Ocelot's Knapsack Calculations for Modeling Power Amplifier and Walsh Code Limits

Kenneth L. Clarkson* John D. Hobby†

June 20, 2007

Abstract

We give a model for the performance impact on wireless systems of the limitations of certain resources, namely, the base-station power amplifier and the available OVSF codes. These limitations are readily modeled in the *loss model* formulation as a *stochastic knapsack*. A simple and well-known recurrence of Kaufman and Roberts allows the predictions of the model to be efficiently calculated. We discuss the assumptions and approximations we have made that allow the use of the model. We have included the model in Ocelot, a Lucent tool for modeling and optimizing cellular phone systems. The model is fast to compute, differentiable with respect to the relevant parameters, and able to model broad ranges of capacity and resource use. These conditions are critical to our application of optimization.

1 Introduction

There are many conditions that reduce the performance of cellular phone systems. Several of these conditions are limitations of shared resources. The theory of *loss model systems* studies the properties of multiple services contending dynamically for a common resource. This note describes the assumptions and approximations the authors have made to apply the loss model formulation to two such common resources: the base-station power amplifier (or *PA*), and the set of OVSF codes used in the forward radio link (from base-station to mobile phone). These resources limit both *circuit* and *packet* services; here we will be mainly concerned with circuit services, with some discussion of how their modeling interacts with the modeling of packet services.

Although the loss model setting is natural and appropriate for this modeling task, it is not a perfect fit. Section 2.1 below gives some assumptions and approximations that we must make to use loss model results to capture the performance effects of the power amplifier and the OVSF codes. For example,

*IBM Research; work was done at Bell Labs

†Bell Labs, Alcatel-Lucent

the power needs of a call can vary continuously, but the simplest loss model results assume resources are measured in discrete units. Some discretization is therefore needed; our approach to that task is outlined in Section 2.2. Next, after giving a more formal description of the loss model calculations for each resource separately, in Section 2.3, we describe the assumptions and approximations that are needed to allow the combined effects of the limitations of these resources, in Section 2.4.

Ocelot also incorporates some performance measures for packet data services. While those measures will be described elsewhere, we still need to make some assumptions in order to combine those measures with the modeling of circuit services; these are outlined in Section 2.5.

Section 3 discusses our overall performance estimate. Section 4 discusses a generalization of the recurrence to situations where multiple resources are shared. The generalization is to expensive to use for optimization, but we can use it as a basis for comparison. Section 5 discusses a simple way to model a system having circuit services with two particular priority types. While not used in Ocelot, such a scheme shows the flexibility of the modeling scheme used. Then Section 6 tests our assumptions and evaluates the accuracy of the performance estimates.

2 Loss Model Systems

Loss model results apply to the following situation. (See, for example, [Kel91, Ros95].)

There is a collection of "jobs," or "calls," contending for a resource; there are M units of the resource, for some M, and there are K distinct kinds of jobs, where job type k requires b_k units of the resource. Once *admitted*, a job uses the resource until the job is done. If admission of a job would raise the total number of units of the resource above M, the job is not admitted: it is *blocked*. It is assumed that jobs arrive at random, and take a random amount of time to be done. We are interested in estimating the *blocking probability* under these conditions, the probability that when a job arrives, all M units of the resource are in use.

We next review the assumptions needed to apply this framework to OVSF codes, and to the power amplifier.

Hereafter, we refer to OVSF codes colloquially as *Walsh* codes, although we mean the more general class of codes.

2.1 Preliminary Assumptions

While a call (that is, job) of a circuit service uses a constant number of Walsh codes over the time the job is in the system, this is only true approximately for PA usage under power control: fading, both slow and fast, results in random variation in the power demands of the call. It is possible to model this variation, but for now we assume that power remains at a fixed level for the duration of a call. Also, power demand is best modeled as varying continuously, so K is infinity; while there are "continuous stochastic knapsack" models of such situations, here we simply discretize power demand.

Approximation 2.1 Power discretization. *Power will be discretized so that M^A units of power, called* bins*, will be available from the PA. The power demands of the calls will be further discretized to use units of $b_i^A = 1, 2, 4 \ldots$ bins.*

Since calls far away from the base-station (that is, at high pathloss) use more power, we cannot assume that the power is fixed for each service; even for voice, some calls may take 1 bin, others 2, others 4, and so on. That is, the appropriate "job type" for analyzing power is not service but what we will call the *power demand class.* As will be apparent from the discussion below, we can regard all calls that use the same power as being the same, as far as analyzing PA usage goes. We don't care what (circuit) service they are, just how much power they use. Section 2.2 has more details on the way the discretization of power demands is done, and Section 2.4 has more details on the relation between services and power demand classes.

The coarse rounding we do is partly justified by a study [Whi] showing that blocking results are relatively insensitive to whether we model calls by different job sizes, or simply by the mean of those sizes; on the other hand, the range of power demands we will consider for 3G services are much greater than for CDMA IS95 voice, and we will be considering a relatively low capacity case: not so many data calls can be supported by a given PA, so it's not clear now that we can simply ignore per-call variations in power demand.

As noted, Walsh codes are already discrete; however, the following approximation will be needed to put them in the loss model framework.

Approximation 2.2 Walsh code additivity. *If the total number of Walsh codes requested is below a given limit, then the requests can be satisfied.*

This is an approximation, because a service needing 2^j Walsh codes will be allocated a set of codes of the form $(k-1)2^j, (k-1)2^j+1, \ldots, k2^j-1$, for some k; it cannot be allocated an arbitrary set of 2^j codes. Moreover, the allocation decision must be done "on-line," as calls arrive.

As our experimental results of Section 6 show, this assumption has a substantial effect; also in the section, we give a "correction" (28) derived empirically, and show experimentally that this change improves the agreement of our calculations with simulation results.

Even the *admission policy* we use for the loss model calculations is an assumption.

Assumption 2.3 Admission. *A call is refused admission (not allowed to connect), when its estimated power demand would raise PA use above M^A bins, or its Walsh code allocation would raise the total number of Walsh codes above a limit M^W.*

As noted above, we are using an additivity approximation for Walsh codes. With respect to the PA, this is apparently the admission policy of some systems, except for the means of obtaining the estimated power demand. As noted above, the estimate we use is static: an expected number of users (Erlangs) obtained from Ocelot's data for the location of users. The actual estimate can be dynamic, and based on the average power demands of the calls of the same service type in service at that moment. Since our model of offered traffic is static, we can't model the changing power requirements of a given call during its duration, and so we can't model any such averaging.

The loss model calculations require one further assumption about the random arrival of jobs.

Assumption 2.4 Poisson arrival. *Jobs arrive as a Poisson process, and their completion time is a random process. If a job is blocked, it goes away (is* cleared*).*

A Poisson process has an associated parameter, its mean λ, and we assume that the completion time has mean $1/\mu$; the key parameter for us is $\rho \equiv \lambda/\mu$, the *load*. We don't need to assume any particular distribution for the completion (or *holding*) times; the calculations are valid with any such holding time distribution.

The load of expected jobs of type k is ρ_k, where $\rho_k = \lambda_k/\mu_k$.

2.2 Continuous versus Discrete Power Demands

In practice, loads are not naturally divided into discrete power demand classes. Instead, we have a series of incremental load contributions with some way of computing power requirements for each. For example, a load contribution $\bar{\rho}$ may require some non-integer number of power bins $\bar{b}$. We cope with this by defining a weighting function w_i for each power demand class i, and contributing $w_i(\bar{b})\bar{\rho}$ to each power demand class i whose weighting function is nonzero at $\bar{b}$.

This is a lot like replacing the single value $\bar{b}$ by a probability distribution centered at $\bar{b}$, as would occur with fading. However, the main purpose here is to cope with the discreteness of the power demands and to ensure that each ρ_i is a smooth function of the $\bar{b}$ values. Hence the weighting functions should have the following properties:

1. Each w_i remains between 0 and 1.

2. At any $\bar{b}$, $\sum_i w_i(\bar{b}) = 1$, so the total of the new loads $\sum_i w_i(\bar{b})\bar{\rho}$ is the original load $\bar{\rho}$.

3. Each w_i must be a C^1 continuous function of $\bar{b}$.

4. At any $\bar{b}$, few $w_i(\bar{b})$ values should be > 0.

5. Unless $\bar{b}$ is too big, $\bar{b} = \sum_i b_i^A w_i(\bar{b})$, so that the total of the new average power demands $\sum_i b_i^A w_i(\bar{b})\bar{\rho}$ is the original average power demand $\bar{\rho}\bar{b}$.

Here "too big" means that $\bar{b}$ is larger than a certain threshold, defined below.

Properties 1 through 4 could also be expressed by saying that the weighting functions are a *partition of unity*. If it weren't for Property 5, we could use nonuniform quadratic B-spline weighting functions. Using cubic splines instead of quadratics as shown in Figure 1 gives the extra degree of freedom needed to ensure Property 5.

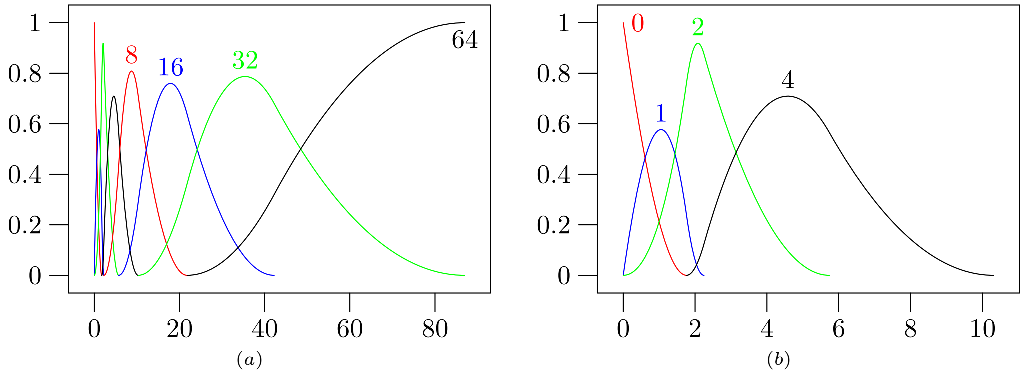


Figure 1: (a) power demand class weighting functions $w_i(\bar{b})$ labeled with the corresponding b_i^A; (b) the first few weighting functions expanded for clarity.

To build the weighting functions, we assume the b_j^A are ordered as

$$0 = b_0^A < b_1^A < b_2^A < \cdots < b_m^A.$$

We use a related set of values $\hat{b}_j$, for $j = 0 \ldots m$, where $\hat{b}_0 = 0$, and choose $\hat{b}_1$, $\hat{b}_2, \ldots, \hat{b}_m$ so that $b_m^A < \hat{b}_m$ and

$$b_i^A < \hat{b}_i < b_{i+1}^A \quad \text{for } 0 < i < m.$$

(For $b_i^A = 0, 1, 2, 4, 8, 16, 32, 64$, Ocelot uses $\hat{b}_i$ values 0, 1.76, 2.25, 5.74, 10.3, 21.8, 42.2, 87.0.)

We need $w_i(\bar{b}) = 0$ and $w_i'(\bar{b}) = 0$ when $\bar{b} \leq \hat{b}_{i-2}$ or $\bar{b} \geq \hat{b}_{i+1}$ so that no more than three w_i functions are ever greater than zero at any $\bar{b}$, and at most two are greater than zero at $\bar{b} = \hat{b}_0, \hat{b}_1, \ldots, \hat{b}_m$. Thus for any $j < m$, applying Properties 2 and 5 with $\bar{b} = \hat{b}_j$ gives two equations

$$w_j(\hat{b}_j) + w_{j+1}(\hat{b}_j) = 1, \tag{1}$$
$$b_j^A w_j(\hat{b}_j) + b_{j+1}^A w_{j+1}(\hat{b}_j) = \hat{b}_j \tag{2}$$

in the two unknowns

$$w_j(\hat{b}_j) \quad \text{and} \quad w_{j+1}(\hat{b}_j).$$

Furthermore, differentiating the equations for Properties 2 and 5 gives two equations

$$w'_j(\hat{b}_j) + w'_{j+1}(\hat{b}_j) = 0, \tag{3}$$
$$b^A_j w'_j(\hat{b}_j) + b^A_{j+1} w'_{j+1}(\hat{b}_j) = 1 \tag{4}$$

in the two unknowns

$$w'_j(\hat{b}_j) \quad \text{and} \quad w'_{j+1}(\hat{b}_j).$$

Thus $w_i(\hat{b}_j)$ and $w'_i(\hat{b}_j)$ can be determined for each $j = 0 \dots m-1$ and each i, recalling that most such values are zero. Since knowing all such values at each cubic segment boundary $\hat{b}_j$ determines all the piecewise-cubic w_i functions, it only remains to choose $w_i(\hat{b}_m)$ and $w'_i(\hat{b}_m)$. This is where $\bar{b}$ is "too big" for Property 5 to apply, so we just have the w_i functions level off with all the weight assigned to b^A_m; i.e., all $w'_i(\bar{b}) = 0$,

$$w_m(\bar{b}) = 1, \quad \text{and} \quad w_i(\bar{b}) = 0 \text{ for } i < m.$$

Note that there is a power demand class $b^A_i = 0$ at $i = 0$. Since this consumes no power resources, the corresponding traffic load can be ignored in much of the following analysis, but it is clearly necessary in order to cope with cases where $\bar{b} < 1$.

2.3 Loss Models

Up to the approximations we have discussed, the loss model systems we have are two instances of a *stochastic knapsack*. The blocking probabilities, and other properties, of the stochastic knapsack can be computed provably and exactly using an efficient calculation, which we next review. This model, and calculation, has seen many applications in network modeling. It has also been applied to model the limitations of wireless systems with respect to reverse-link interference,[JHR02] an area in which we have not applied it.

The loss model framework, again, is the following. We have a collection of K kinds of jobs contending for a resource of M units, and job type k needs b_k units. Jobs arrive Poisson, with load ρ_k. For the Walsh codes, the job type is the type of service; for the PA, the job type is the power demand class.

The stochastic knapsack calculation (Kaufman-Roberts recurrence) allows us to find the steady-state probability distribution of the number of resources in use, and the blocking probabilities per job type.[Kau81, Rob81, Ros95] A derivation of this calculation is outlined, in a more general setting, in Section 4 below.

Let $g(c)$ satisfy $g(c) = 0$ for $c < 0$ or $c > M$, and $g(0) = 1$, and let

$$g(c) = \frac{1}{c} \sum_k \rho_k b_k g(c - b_k) \tag{5}$$

for $c = 1 \dots M$. Then the steady-state probability that c resource units are used is $\hat{g}(c) \equiv g(c)/G$, where

$$G \equiv \sum_{c} g(c). \tag{6}$$

The blocking probability for a circuit service needing b_k units is then

$$R_k \equiv \sum_{c<b_k} \hat{g}(M-c). \tag{7}$$

Equivalently, let

$$G(c) \equiv \sum_{c' \le c} g(c').$$

Then

$$R_k = 1 - G(M-b_k)/G(M). \tag{8}$$

It will be convenient to define

$$P_k \equiv 1 - R_k = G(M-b_k)/G(M) \tag{9}$$

as the "passing probability".

We will use the recurrence in computing the passing probability, first for Walsh codes and then for the power amplifier.

2.4 Assumptions for Integrated Analysis

So far, discussion has been about analyzing the PA blocking probability in isolation, and similarly the Walsh code blocking probability, in isolation. Moreover, discussion has not addressed packet data QoS analysis. This subsection discusses the approximations and assumptions we make to analyze the joint effect of the Walsh code and PA blocking, and to integrate packet data analysis, and other remaining approximation.

We model the combined effect of PA limitations and Walsh limitations sequentially, for non-packet services:

Approximation 2.5 Cascade model. *We will first compute the blocking per service due to Walsh code limitations, and then compute the blocking due to the PA of the resulting reduced load, on a service-by-service basis.*

That is, high blocking of a given service by Walsh code limitations implies a reduced load when considering that service with respect to PA limitations. The reduction in PA load for that service is assumed to be uniform across different power demand classes for that service. (This is expressed symbolically as (16) below.)

This is an approximation, because even when the PA is highly loaded, and a service might be blocked as a result, we will still consider the service at full load for the Walsh code calculation.

A more sophisticated approach is taken with *reduced-load approximation*, also known as *Erlang fixed-point approximation*. In that approach, as specialized to the task here, the load for computing the blocking due to Walsh codes would be reduced by considering the blocking due to the PA, as well as the other way around. That is, mutually consistent blocking probabilities and reduced loads would hold for Walsh codes and the PA simultaneously. There is always a unique solution for such a set of conditions, for the $K = 1$ case,[Whi85, Kel86] and always at least one solution for $K \geq 1$.[CR93]

Although a reduced load approximation might be an improvement in accuracy, we don't take that approach here. Such an approximation is typically found by fixed-point iteration, which here would amount to substituting the output load back into the Walsh code blocking probability calculation, and then using the original load times the Walsh blocking to determine the input load for the PA. Doing this repeatedly is generally thought to converge rapidly, but it is still much slower than the simple cascade. (Although a simplified loss model calculation may make it feasible.[TM96]) Moreover, unless the fixed-point computation is done to machine accuracy, it is difficult to determine the derivative of the output of such an approximation; the derivative is very useful for optimization. Another possibility would be to iterate a small, fixed, number of times. However, the derivative calculation would then be intractable.

Assumption 2.6 Equal priority circuit service. *All circuit services (including voice and data) have equal priority.*

That is, we don't model a policy where circuit data services are thrown off in overload. This assumption can be avoided: in Section 5, we discuss how to compute the blocking probabilities efficiently for two priority classes, one of which is "best effort"; however, this approach seems hard to apply in combination with calculation of packet data QoS estimates.

Approximation 2.7 Activity factors. *In addition to loads, we also have* activity factors *that specify another form of variation in the use of resources.*

The activity factor models the use of the resource during a job, giving an indication of the *average* use of the resource. The role of activity factors is different for different services and for the two resources. While an inactive voice or circuit call uses less (or no) power, it does use its allocated Walsh codes, so the "activity factor" for circuit Walsh code usage is one, even if the general activity factor for PA usage is less than one. We will model the situation by reducing the usage estimate b_k^A for the PA by the activity factor, but not doing so for Walsh codes. An inactive packet data call uses neither its Walsh codes nor the PA, so we explicitly include the activity factor into the M/M/m queue calculation for Walsh codes and 3G1X PA usage. Since the UMTS PA performance values for packet data are frozen, the activity factor for a UMTS packet data service is frozen.

2.5 Packet Data

Packet data services do not satisfy, even approximately, the assumptions of the stochastic knapsack calculation. From our assumptions, however, the results of the stochastic knapsack calculation for the circuit services can be used to more accurately predict the system performance for packet data services.

Assumption 2.8 Packet doesn't affect circuit. *The circuit services affect the resources available to the packet services, but* not *the other way around.*

The basis of this assumption is that one simple mechanism for handling overload, that has been implemented, is to disconnect data services when the PA is in an "overloaded" state. Such a state occurs where the power the PA is supplying is above its nominal rating. We assume a similar condition holds for Walsh codes. This assumption allows us to model the PA use by first applying "loss model" calculations to the circuit services, and then finding the quality of service available to the packet services.

Approximation 2.9 Packet service quality function. *For given power available from the PA, all users of a packet data service will see the same performance, which is a function of the available power and of the average power needed per call by the users of that packet service.*

From this assumption, we model the expected QoS for a packet-data service as follows: we use a quality function $P_p(X, Z)$, taking values in $0 \dots 1$, where X is the expected number of packet users, and Z is the ratio of available units of the resource to the average need of that resource by a user. (So Z is the "number of channels".) The function $P_p()$ is analogous to P_k for circuit services: we want $P_p()$ as large as possible. The expected quality of service for packet-data services is then estimated as

$$\bar{P_p} \equiv \sum_c \hat{g}(c) P_p(\hat{\rho}_p, \hat{\rho}_p(M-c)/D_p), \tag{10}$$

where $\hat{\rho}_p$ is the expected number of packet users and D_p is the expected total demand for the resource by the packet user. If we define $\tilde{\rho}_k$ as the expected number of packet data users for the job type k, then

$$\hat{\rho}_p = \sum_k \tilde{\rho}_k = \sum_{i,j \notin C} \rho_{ij} \tag{11}$$

and

$$D_p \equiv \sum_k b_k \tilde{\rho}_k \tag{12}$$

So $D_p/\hat{\rho}_p$ is an estimate of the average resource need per packet-data user, and again, $\hat{g}(c) \equiv g(c)/G$ is the steady-state probability that c units are in use by circuit services, so that $M - c$ are available for packet data.

The computation of $P_p(X, Z)$ is outside the scope of this paper, but here is an overview. To compute $P_p(X, Z)$, we use the "normalized relative throughput" (a function of delay) from the Walsh code calculations, and then from the PA calculations, and take the minimum of the two throughputs, or equivalently, compute the normalized relative throughput corresponding to the maximum delay due to Walsh code and PA limitations. The throughput calculations are based, for the PA in UMTS, on Monte Carlo simulations;[Gra] the results of these simulations are tabulated and used to generate a smooth function in Ocelot. For the PA in 3G1X, *and* for the Walsh codes in all technologies, we will use a throughput estimate based on bounded M/M/m queues. [1]

3 Performance Estimate

First we describe the loss model calculations, and then the overall performance estimate.

3.1 Per-service Performance

As described above, we will do a loss model calculation for the Walsh codes, and then use the resulting reduced load to do a loss model calculation for the PA, and use the results of those calculations to compute performance estimates for packet data services.

Each service j, such as voice, circuit data, packet data, etc., will be modeled as having offered load ρ_{ij} for power demand b_i^A and Walsh code usage b_j^W; that is, ρ_{ij} expected users using service j will need b_i^A bins of the PA and b_j^W Walsh codes.

The load ρ_j^W of circuit service j for the Walsh-code loss-model calculation is

$$\rho_j^W \equiv \sum_i \rho_{ij}, \tag{13}$$

and we can take ρ_j^W to be zero for $j \notin C$, where C is the set of indices of circuit services (that is, voice or circuit data). Having done the loss model calculation for the Walsh codes, we have Walsh code usage probabilities $\hat{g}^W(c)$ and passing probabilities P_j^W. As discussed above, we use the $\hat{g}^W(c)$ values to compute $\bar{P_p}^W$, the normalized relative throughput of packet data services due to Walsh code limitations. Here, for packet data service j, the load for service j (job type j) is

$$\tilde{\rho}_j^W = \sum_i \rho_{ij},$$

[1] The PA model using bounded M/M/m queues was suggested by Harish Viswanathan and John Graybeal, and the use of the performance measure of normalized relative throughput was suggested by Graybeal.

and $\tilde{\rho}_j^W = 0$ if $j \in C$. This implies

$$\hat{\rho}_p^W = \sum_j \tilde{\rho}_j^W = \sum_{i,j \notin C} \rho_{ij} \tag{14}$$

and

$$D_p^W \equiv \sum_{j \notin C} b_j^W \tilde{\rho}_j = \sum_{i,j \notin C} b_j^W \rho_{ij} \tag{15}$$

For the PA loss-model calculation, the load values ρ_i^A are

$$\rho_i^A \equiv \sum_{j \in C} P_j^W \rho_{ij}. \tag{16}$$

Together with the bin requirements b_i^A, these yield PA bin usage probabilities $\hat{g}^A(c)$ and passing probabilities P_j^A. The usage probabilities are used to compute $\bar{P}_p{}^A$, the normalized relative throughput of packet data services due to PA limitations. Here packet data load for power demand class (PA job type) i is

$$\tilde{\rho}_i^A = \sum_{j \notin C} \rho_{ij}.$$

Note that since our model of QoS for packet data is based on delay, there is no modeled reduction of PA demand by packet calls due to Walsh code limitations. We have

$$\hat{\rho}_p^A = \sum_i \tilde{\rho}_i^A = \sum_{i,j \notin C} \rho_{ij}, \tag{17}$$

so indeed $\hat{\rho}_p^A = \hat{\rho}_p^W = \hat{\rho}_p$, and

$$D_p^A \equiv \sum_i b_i^A \tilde{\rho}_i = \sum_{i,j \notin C} b_i^A \rho_{ij} \tag{18}$$

3.2 Overall Performance

We can now join together the blocking probabilities and performance estimates to obtain an overall performance estimate.

We merge together the packet data estimates $\bar{P}_p{}^W$ and $\bar{P}_p{}^A$ using a "smooth min" function $\mathcal{M}(.,.)$ to obtain a packet performance estimate

$$\bar{P}_p{}^O \equiv \mathcal{M}(\bar{P}_p{}^W, \bar{P}_p{}^A). \tag{19}$$

If we used the usual min instead of the smooth min, the estimate would not be differentiable everywhere, and a derivative discontinuity would interfere with optimization.

Let $\mathcal{I}_j$ be a weighting factor indicating the "importance" of service j. We will combine the estimates together by weighting using $\mathcal{I}_j$ and using the appropriate loads. Let

$$\mathcal{L}_p \equiv \sum_{j \notin C} \sum_i \mathcal{I}_j \rho_{ij}.$$

Our measure of overall performance is $T/\mathcal{L}$, where

$$T \equiv \sum_{j \notin C} \sum_i \mathcal{I}_j \rho_{ij} \bar{P_p}^O + \sum_{j \in C} \sum_i \mathcal{I}_j \rho_{ij} P_j^W P_i^A = \mathcal{L}_p \bar{P}_p + \sum_{j \in C} \sum_i \mathcal{I}_j \rho_{ij} P_j^W P_i^A \tag{20}$$

and

$$\mathcal{L} \equiv \sum_j \sum_i \mathcal{I}_j \rho_{ij}.$$

This completes the description of the calculations, except for computing derivatives.

4 Loss Models for Multiple Resources

While the Kaufman-Roberts recurrence has been applied to the sharing of a single resource, it can be extended to model the sharing of multiple resources. Next we give such an extension, which can then be compared with the results of the "cascade" method of combining Walsh code and PA results.

The extension is appropriate when there there are two or more resources, and the units b_{rk} of resource r for job-type k may be different for each r. Also the total number of total units M_r of each resource r may be different. Here a multi-dimensional recurrence, like Kaufman-Roberts, can be used. As noted, this recurrence is expensive: the time needed to evaluate it is $\Omega(K \prod_i M_r)$ in the worst case. This is too slow to use routinely in applications, but it allows the exact blocking probability to be computed for comparisons with approximations.

The Kaufman-Roberts recurrence has been extended before, in the context of *loss networks*, [Ros95], where the resources being shared are links in a network. The situation here is a little different, because the b_{rk} values can be different for different resources r, and the sharing is less complicated: every job may use each one of the resources. (However, the discussion here cannot be regarded as new, and is included mainly for completeness.) In the loss network setting, the latter can be modeled as a single path; however, in a loss network, effectively $b_{rk} = b_{r'k}$ for all r and r', and so modeling the single path is simply a matter of using the link with the smallest M_r. Finally, in a loss network, the existence of disjoint paths (or nearly disjoint paths) implies the plausibility of the approximation of assuming that resource blocking probabilities are independent. Such an approximation is clearly inappropriate here.

We derive the recurrence analogously to the derivation given in Ross[Ros95]. Let the jobs being done as a given moment be given as column K-vector $\mathbf{n} = (n_1, \ldots, n_K)$, where n_k is the number of jobs of type k. Let $\mathbf{B}$ denote the $R \times K$ matrix of values b_{rk}, the number of units of resource r used by job type k. Let $b_{\cdot k}$ and $b_{r\cdot}$ denote the k'th column and r'th row of $\mathbf{B}$. Let $\mathbf{c}$ denote the column R-vector of units of resources in use, where $\mathbf{c}_r$ denotes the units of resource r in use. Let $\mathcal{S}(\mathbf{c})$ denote the set of vectors $\mathbf{n}$ of job types that use resources $\mathbf{c}$, so

$$\mathcal{S}(\mathbf{c}) \equiv \{\mathbf{n} \mid \mathbf{B}\mathbf{n} = \mathbf{c}\}.$$

Let $\pi(\mathbf{n})$ denote the steady-state probability of job-type distribution $\mathbf{n}$, in the unbounded case, that is, where all resources are infinite. There is an explicit expression for $\pi(\mathbf{n})$, but the only property we will need here is the balancing equation

$$n_k \pi(\mathbf{n}) = \rho_k \pi(\mathbf{n} - e_k), \tag{21}$$

where e_k is the K-vector that is one in entry k and zero elsewhere. This expression results from the fact that, in the steady state, the probability of a transition from state $\mathbf{n}$ to state $\mathbf{n} - e_k$ is equal to the probability of a transition in the other direction. We can write relation (21) as

$$\mathbf{n}\pi(\mathbf{n}) = \boldsymbol{\rho}\boldsymbol{\pi}(\mathbf{n}), \tag{22}$$

where $\boldsymbol{\rho}$ is the diagonal $K \times K$ matrix with entries $\boldsymbol{\rho}_{kk} \equiv \rho_k$, and $\boldsymbol{\pi}(\mathbf{n})$ is the $K \times 1$ vector with $\boldsymbol{\pi}(\mathbf{n})_k \equiv \pi(\mathbf{n} - e_k)$ if $n_k > 0$, and $\boldsymbol{\pi}(\mathbf{n})_k = 0$ when $n_k = 0$.

Let $q(\mathbf{c})$ denote the steady state probability of resource distribution $\mathbf{c}$ in the unbounded case, so

$$q(\mathbf{c}) = \sum_{\mathbf{n} \in \mathcal{S}(\mathbf{c})} \pi(\mathbf{n}), \tag{23}$$

and let $\mathbf{q}(\mathbf{c})$ denote the $K \times 1$ vector with

$$\mathbf{q}(\mathbf{c})_k \equiv q(\mathbf{c} - b_{\cdot k})$$

when $\mathbf{c} \geq b_{\cdot k}$, and $\mathbf{q}(\mathbf{c})_k = 0$ otherwise.

Another simple statement about $\boldsymbol{\pi}(n)$ follows from the fact that $\mathbf{n} \in \mathcal{S}(\mathbf{c})$ if and only if $\mathbf{n} - e_k \in \mathcal{S}(\mathbf{c} - b_{\cdot k})$, when $\mathbf{n}_k > 0$. From this fact, and the definitions, it follows that

$$\begin{aligned}
\sum_{\mathbf{n} \in \mathcal{S}(\mathbf{c})} \boldsymbol{\pi}(\mathbf{n})_k &= \sum_{\mathbf{n} \in \mathcal{S}(\mathbf{c})} \pi(\mathbf{n} - e_k) \\
&= \sum_{\mathbf{n} \in \mathcal{S}(\mathbf{c} - b_{\cdot k})} \pi(\mathbf{n}) \\
&= q(\mathbf{c} - b_{\cdot k}) = \mathbf{q}(\mathbf{c})_k,
\end{aligned}$$

that is,

$$\mathbf{q}(\mathbf{c}) = \sum_{\mathbf{n} \in \mathcal{S}(\mathbf{c})} \boldsymbol{\pi}(\mathbf{n}). \tag{24}$$

(cf. (23))

A recurrence relation for $q(\mathbf{c})$ can now be derived, using (22) and (24). We have

$$\begin{aligned}
\mathbf{c}q(\mathbf{c}) &= \sum_{\mathbf{n}\in\mathcal{S}(\mathbf{c})} \mathbf{c}\pi(\mathbf{n}) = \sum_{\mathbf{n}\in\mathcal{S}(\mathbf{c})} (\mathbf{B}\mathbf{n})\pi(\mathbf{n}) \\
&= \mathbf{B}\sum_{\mathbf{n}\in\mathcal{S}(\mathbf{c})} \mathbf{n}\pi(\mathbf{n}) = \mathbf{B}\sum_{\mathbf{n}\in\mathcal{S}(\mathbf{c})} \boldsymbol{\rho}\boldsymbol{\pi}(\mathbf{n}) \\
&= \mathbf{B}\boldsymbol{\rho}\sum_{\mathbf{n}\in\mathcal{S}(\mathbf{c})} \boldsymbol{\pi}(\mathbf{n}) \\
&= \mathbf{B}\boldsymbol{\rho}\mathbf{q}(\mathbf{c}).
\end{aligned}$$

This reasoning gives us R expressions for $q(\mathbf{c})$, which are all equal: the matrix-vector expression above implies that for each $r = 1 \ldots R$,

$$q(\mathbf{c}) = \frac{1}{c_r} b_{r\cdot}\boldsymbol{\rho}\mathbf{q}(\mathbf{c}).$$

These dependencies are a consequence of the linear dependency $\mathbf{c} = \mathbf{B}\mathbf{n}$.

We have a recurrence for $q(\mathbf{c})$, the steady-state probability of having the units of each resource in use be $\mathbf{c}$, in the unbounded case. How does this help us find the corresponding probabilities for the bounded case, where we must have

$$\mathbf{c} = \mathbf{B}\mathbf{n} \leq \mathbf{M},$$

where $\mathbf{M}$ is the R-vector of bounds on the resources? The remarkable fact is that the bounded probabilities are proportional to the unbounded probabilities, so we can set up and solve the analogous recurrence for $g(\mathbf{c})$, with initial, boundary values set at one, and then compute $Q(\mathbf{c}) = g(\mathbf{c})/G$ for every $\mathbf{c}$, where G is a "normalization constant" equal to the sum of the $g(\mathbf{c})$ values for all $\mathbf{c}$ with $\mathbf{c} \leq \mathbf{M}$. The validity of this calculation is a consequence of the reversibility of the Markov chain with state space $\{\mathbf{n} \mid \mathbf{n} >= 0\}$.[Ros95]

It follows that we can solve the recurrence above, starting with $q(\mathbf{0}) = 1$, for all $\mathbf{c} \leq \mathbf{M}$, and then normalize by $\sum_{\mathbf{c}\leq\mathbf{M}} q(\mathbf{c})$.

An obvious problem here is the expense of solving the recurrence, when R is not small. One helpful possibility is that many values of $q(\mathbf{c})$ may be zero, so that if we solve the recurrence so that one nonzero entries are touched, the calculation will be correspondingly sped up.

5 Two-Priority Systems

As given above as Assumption 2.6, we assume that all circuit services have the same priority. It is not unusual, however, to have circuit data services at lower priority than voice services, where the data services are thrown off in overload. In such a case, the higher priority (voice) services are unaffected by the lower priority (data) services, while the data services make do with whatever resources are left over from the demands of the higher priority services. Here we note that it is possible to model such a condition with the stochastic knapsack, without a significant computational cost. The idea is similar to what is done to include the modeling of packet data (and is mutually exclusive with that modeling). The loss model calculation (5) is done as before, but omitting the circuit data services, yielding functions $g^V(c)$, $\hat{g}^V(c)$, and $G^V(c)$. The circuit data services would have a separate calculation, leading to functions $g^D(c)$ and $G^D(c)$ for the circuit data services alone, and a function $R_k^D(c)$, where analogously to (8),

$$R_k^D(c) \equiv 1 - G^D(c - b_k)/G^D(c),$$

the blocking probability for (data) job type k, given that c resource units are available. The blocking probability for a lower priority circuit data service is then

$$\sum_c \hat{g}^V(c) R_k^D(M - c),$$

the probability that c resource units are used by the higher priority voice services, times the blocking for data, given that $M - c$ units are available. This requires only a constant factor more time to compute, but does preclude the convolution that is done for packet data.

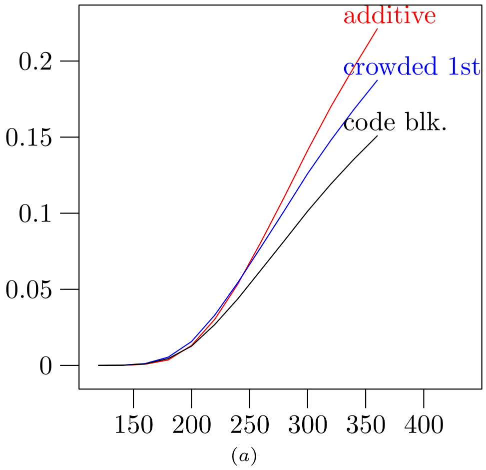
6 Experimental Studies

Since it is hard to compare our model directly with the real world, we built a Monte Carlo simulator that models call arrivals and departures, Walsh code allocation and deallocation, and blocking due to Walsh and power limitations. It does not operate in discrete time steps, but rather uses exponentially-distributed random variables to decide when the next event happens. Each simulation ran for at least 30,000 steps with one half or one third of that reserved for "warm-up time" not used in gathering statistics. Furthermore, each statistic reported below is averaged over at least 10 such simulation runs.

6.1 Walsh Code Additivity

Running the Monte Carlo simulator with the available power M^A set very high allows us to compare Approximation 2.2 (Walsh code additivity) to the popular *crowded-first* allocation scheme [YCT01, RS02], also called *crowded-first-code* [CTW03]. One would naturally expect the additivity assumption to be optimistic with respect to the overall blocking probability, because it amounts to assuming that there is never any *code blocking* (blocking calls due to otherwise sufficient free Walsh codes not forming a large enough contiguous block). However, Figure 2 shows that the assumption actually becomes *pessimistic* at high loads. Here code blocking affects more calls that demand many Walsh codes. For a lower overall blocking, it is good strategy to block such "large jobs" unnecessarily if this is likely to prevent many small jobs from being blocked.

When the distribution of Walsh code demands is skewed so there are fewer large jobs and more small jobs, Figure 3 shows that the additivity assumption is more pessimistic at high loads, and the crossover from optimism to pessimism



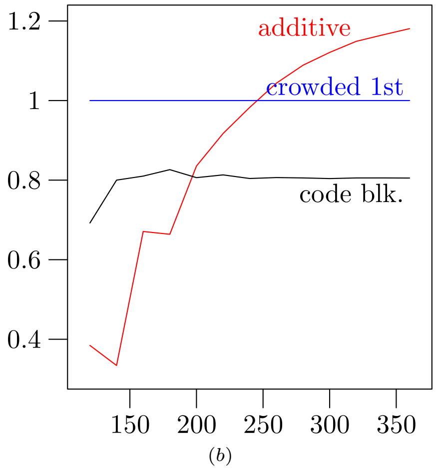


Figure 2: (a) Blocking versus average Walsh code demand ℓ under the additivity assumption and crowded-first allocation, with a separate graph for the code blocking component; (b) the ratio of each type of blocking to crowded-first blocking. Each call is equally likely to demand 1, 2, 4 or 8 Walsh codes.

happens at a lower load and a lower blocking rate. This crossover tends to be at blocking rates near a few percent, which is a reasonable operating point for a loaded cell phone system.

To get a better idea of what Walsh code demand qualifies a call as "large," note that each value for the total free space σ in the Walsh tree leads to a probability distribution for how often the maximum contiguous block of available Walsh codes is 1, 2, 4, 8, 16, etc. Figure 4 shows

$$\gamma := 2^{E[\lg \beta | \sigma]}$$

as a function of σ, where β is the maximum contiguous available Walsh size. Here $E[\lg \beta \mid \sigma]$ is the conditional expectation of $\lg \beta$ given σ, for some specific ρ and b^W values, and $\lg n := \log_2 n$.

Figure 4 suggests that γ is roughly linear in σ, but depends on the Walsh code demands of incoming calls, and is bounded by the largest power of 2 not exceeding σ. Thus an empirical estimate for γ can be of the form,

$$\gamma \approx \min(3 + f(z, \ell) \cdot \sigma,\ 2^{\lfloor \lg \sigma \rfloor}), \tag{25}$$

where $\ell := \sum_j \rho_j^W b_j^W$ and $z := \sqrt{\sum_j \rho_j^W (b_j^W)^2 / \sum_j \rho_j^W}$, the RMS mean of the Walsh demands b_j^W weighted by the associated traffic ρ_j^W. Since it can be useful to be able to compute estimated γ values, we shall use the rather arbitrary empirical formula

$$f(z, \ell) = \frac{7.5 + 8.53z + 0.157z^2}{\ell + \max(0, 24z - 100)}. \tag{26}$$

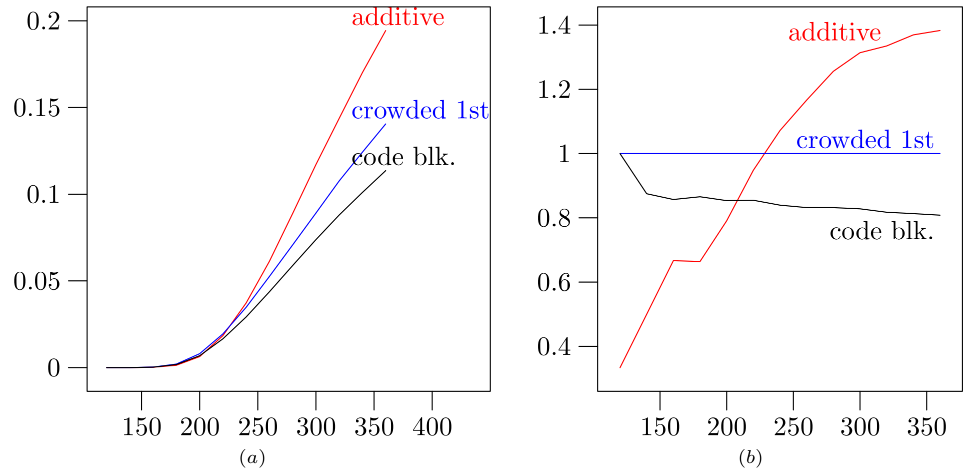


Figure 3: (a) Blocking versus average Walsh code demand ℓ under the additivity assumption and crowded first allocation, with a separate graph for the code blocking component; (b) The ratio of each type of blocking to crowded-first blocking. Relative likelihoods for Walsh code demands 1, 2, 4, 8 are 1, 0.6, 0.36, 0.216.

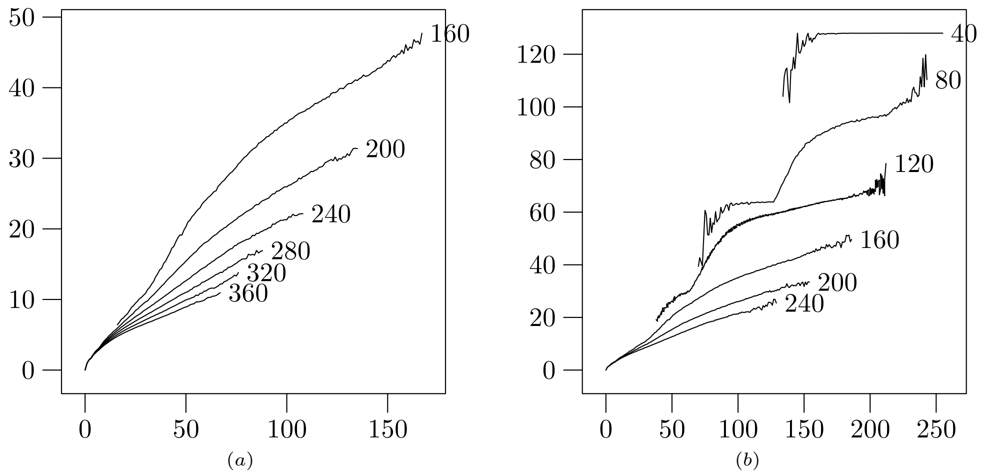


Figure 4: Graphs of maximum Walsh codes per call γ versus total Walsh tree free space σ for RMS mean Walsh codes per call $z = 4.61$ and for various total Walsh loads ℓ. (a) shows only σ values with ≥ 1000 occurrences in the simulator runs; (b) uses a lower limit of 10 occurrences for $\ell = 40, 80$ and a limit of 100 for larger ℓ values.

How much does it help to base call blocking on (25) and (26) instead of just using the additivity assumption? This seems promising in simulator runs; i.e., Figure 5 shows that it does give a much better match to crowded-first blocking for the examples of Figures 2b and 3b.

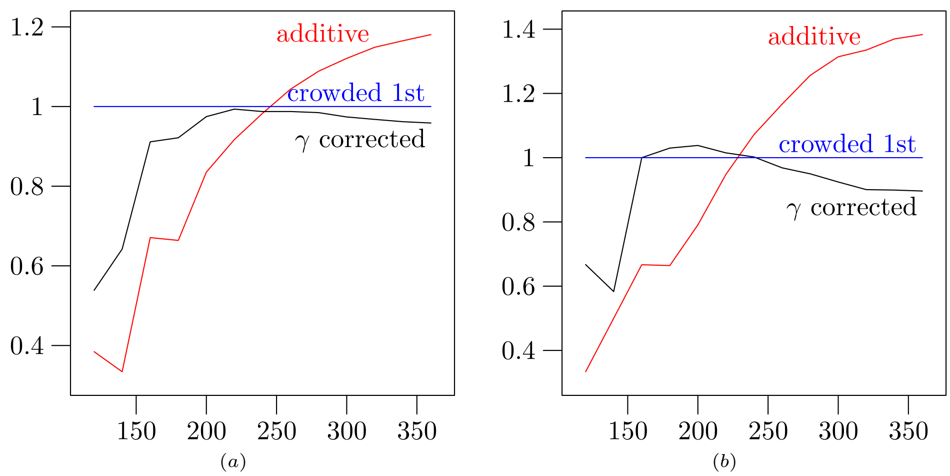


Figure 5: Graphs of the ratio of blocking from the additive assumption to crowded first blocking versus Walsh demand ℓ, including a version corrected via (25). For (a), Walsh demands 1, 2, 4, 8 are equally likely; for (b), their relative likelihoods are 1, 0.6, 0.36, 0.216.

This can also be solved numerically if one is willing to solve a large sparse linear system, where there is one variable for each system configuration, and the system configuration depends on the number of calls using each Walsh code size b_j^W. If, for instance, $M^W = 128$ and the Walsh demands b_j^W are $\{1, 2, 4, 8\}$, then there is one probability to compute for each of the 222,241 ways of choosing non-negative integers q_0, q_1, q_2, q_3 such that $q_0 + 2q_1 + 4q_2 + 8q_3 \leq 128$. The problem is that basing the blocking on (25) causes the resulting probability distribution for $c = q_0 + 2q_1 + 4q_2 + 8q_3$ to disobey the Kaufman-Roberts recurrence.

More heuristically, using ℓ and z as defined above and $\sigma(c) = M^W - c$, the Kaufmann-Roberts recurrence can be modifed to

$$g(c) = \frac{1}{c} \sum_k \rho_k^W b_k^W g(c - b_k^W) \delta_{c-b_k,k}, \tag{27}$$

where

$$\delta_{c,k} = \begin{cases} 1 & \text{if } b_k^W < \min\{3 + f(z,\ell) \cdot \sigma(c),\, 2^{\lfloor \lg \sigma(c) \rfloor};\} \\ \frac{1}{2} & \text{if } b_k^W = \min\{3 + f(z,\ell) \cdot \sigma(c),\, 2^{\lfloor \lg \sigma(c) \rfloor};\} \\ 0 & \text{otherwise.} \end{cases} \tag{28}$$

Figure 6 gives an example of the resulting $g(c)$ distribution for a scenario where it is much more accurate than the unmodified Kaufman-Roberts solution for the high c values that are relevant for computing blocking probability. See also Figure 8, discussed just below.

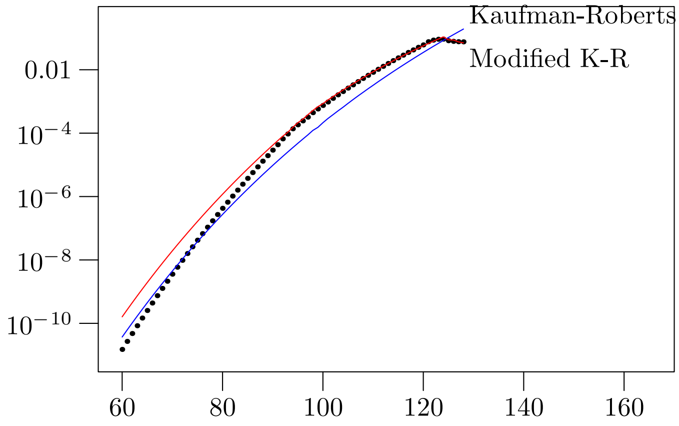


Figure 6: The solution to (27) as a function of Walsh codes in use c along with the corresponding probability from Kaufman-Roberts. The dotted line gives the distribution from solving from solving the large sparse linear system that describes blocking based on (25). This is for the Walsh code demands from Figure 5 with $M^W = 128$ and $\ell = 268.4$.

6.2 The cascade model

How good is the cascade model (Approximation 2.5)? One would expect it to perform well when Walsh limitations strongly predominate over power limitations. This is demonstrated by the scatter plots in Figure 7. These show the model's prediction for overall performance, versus corresponding simulator results. Here there were two services and two power demand classes with

$$M^W = 64, \quad M^A = 100, \quad cI_1, \mathcal{I}_2 = 1, 2, \quad b_1^W, b_2^W = 16, 1, \quad b_1^A, b_2^A = 1, 4,$$

and 81 different problem instances were obtained by trying all possible load matrices where

$$\rho_{1,1}, \rho_{2,1} \in \{2, 8, 32\}, \quad \rho_{1,2}, \rho_{2,2} \in \{1, 4, 16\}. \tag{29}$$

Since Figure 7b shows such a close match between the cascade model and the simulator results, almost all the disagreement in Figure 7a must be due to the Walsh additivity assumption. This conclusion is bolstered by Figure 8, which compares the performance estimated using the empirical correction (28) with simulator results, under the same conditions. This shows a clear improvement relative to Figure 7a. When using the empirical correction, the blocking probability calculation is slightly more complicated: rather than $\sum_{c>M^W-b_k} g(c)/\sum_c g(c)$, it is

$$\frac{\sum_c g(c)(1-\delta_{c,k})}{\sum_c g(c)}.$$

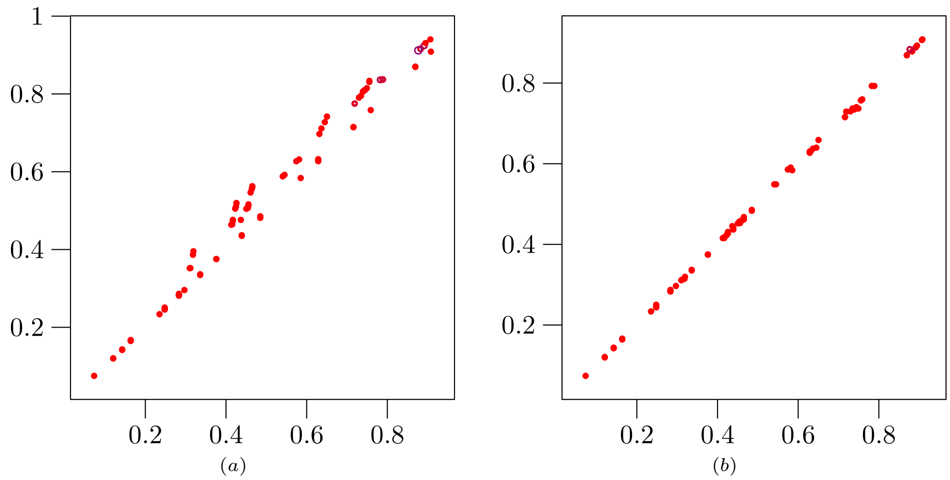


Figure 7: (a) Overall performance predicted by the Monte Carlo simulator for various Walsh-limited scenarios versus the corresponding $T/\mathcal{L}$ from the cascade model; (b) the same except with the simulator using Walsh additivity assumption for admission control.

Now consider 81 power-limited scenarios where the 2 services and 2 power demand classes have

$$M^W = 256, \quad M^A = 100, \quad cI_1, \mathcal{I}_2 = 1, 2, \quad b_1^W, b_2^W = 8, 1, \quad b_1^A, b_2^A = 11, 4,$$

and (29) gives 81 sets of $\rho_{i,j}$ values. In this case the Walsh-additivity assumption does not matter and we get good agreement between simulator results and the cascade model as shown in Figure 9.

Tweaking the scenarios so that

$$M^W = 64, \quad M^A = 100, \quad cI_1, \mathcal{I}_2 = 1, 2, \quad b_1^W, b_2^W = 8, 1, \quad b_1^A, b_2^A = 8, 3,$$

while still using the 81 sets of $\rho_{i,j}$ values from (29) gives Figure 10. Many of these are scenarios where Walsh and power limitations both matter, yet there isn't a lot of scatter in the figure, and the small circles for scenarios where both limitations matter do not appear particularly problematical. In fact, comparing 10a to 10b shows more scatter due to the Walsh additivity assumption than due to the cascade model.

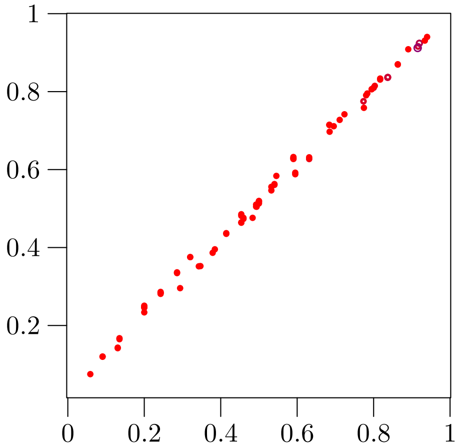


Figure 8: Overall performance predicted by the Monte Carlo simulator for various Walsh-limited scenarios versus the corresponding $T/\mathcal{L}$ from the cascade model, using empirical correction;

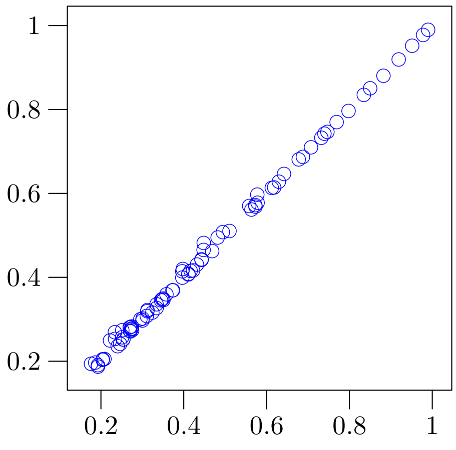


Figure 9: Overall performance predicted by the Monte Carlo simulator for various power-limited scenarios versus the corresponding $T/\mathcal{L}$ from the cascade model.

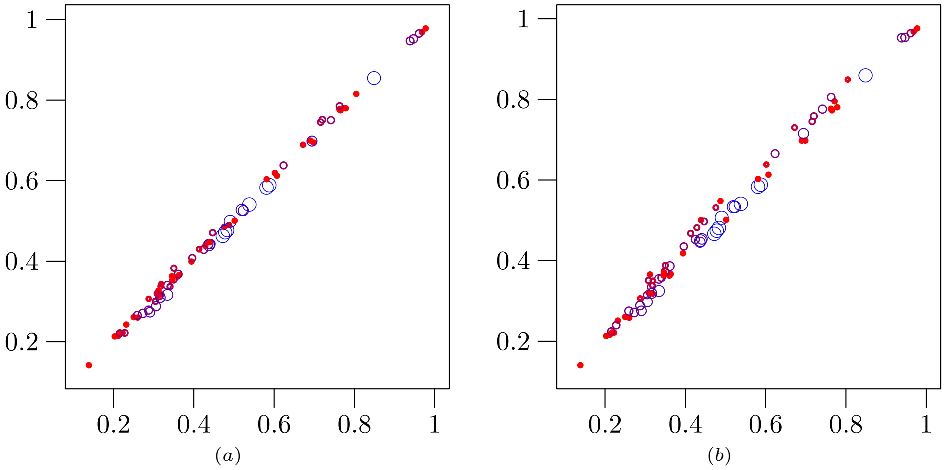


Figure 10: (a) Overall performance predicted by the Monte Carlo simulator for various scenarios versus the corresponding $T/\mathcal{L}$ from the cascade model without the empirical correction; (b) the same except with the simulator using Walsh additivity assumption for admission control. Dots like those in Figure 7 denote Walsh-limited scenarios; circles like those in Figure 9 denote power-limited scenarios; and anything in-between denotes a scenario where both limitations matter.

References

[CR93] S.-P. Chung and K. W. Ross. Reduced load approximations for multirate loss networks. *IEEE Trans. Communications*, 41(8):1222–1231, Aug. 1993.

[CTW03] C.-M. Chao, Y.-C. Tseng, and L.-C. Wang. Reducing internal and external fragmentations of OVSF codes in WCDMA systems with multiple codes. *Wireless Communications and Networking*, 1:693–698, March 2003.

[Gra] J. Graybeal. Monte Carlo simulation of UMTS packet data performance. Personal communication.

[JHR02] M. Jaber, S. A. Hussain, and A. Rouz. Modified stochastic knapsack for UMTS capacity analysis. *FUJITSU Sci. Tech. J.*, 38(2):183–191, Dec. 2002.

[Kau81] J. S. Kaufman. Blocking in a shared resource environment. *IEEE Trans. Commun.*, 29:1474–1481, Aug. 1981.

[Kel86] F. P. Kelly. Blocking probabilities in large circuit switched networks. *Adv. Appl. Probabil.*, 18:473–505, 1986.

[Kel91] F. P. Kelly. Loss networks. *Ann. Appl. Prob.*, 1:319–378, 1991.

[Rob81] J. W. Roberts. *A service system with heterogeneous user requirements*, pages 423–431. North-Holland, 1981.

[Ros95] K. W. Ross. *Multiservice loss models for broadband telecommunication networks*. Springer, 1995.

[RS02] A. N. Rouskas and D. N. Skoutas. OVSF codes assignment and reassignment at the forward link of W-CDMA 3G systems. In *Proc. of IEEE PIMRC*, volume 5, pages 2404–2408, 2002.

[TM96] F Théberge and R. Mazumdar. An efficient reduced load heuristic for computing call blocking in large multirate loss networks. In *Global Telecommunications Conference, 1996. GLOBECOM '96. 'Communications: The Key to Global Prosperity*, volume 1, pages 6–10, Nov. 1996.

[Whi] P. Whiting. Personal communication.

[Whi85] W. Whitt. Blocking when service is required from several facilities simultaneously. *AT&T Tech. J.*, 64(8):18071857, 1985.

[YCT01] C.-M. Chao Y.-C. Tseng. Code placement and replacement strategies for wideband cdma ovsf code tree management. In *Proc. of IEEE GLOBECOM*, volume 1, pages 562–566, 2001.