1985

On a Recurrence Equation Arising in the Analysis of Conflict Resolution Algorithms

Wojciech Szpankowski
Purdue University, spa@cs.purdue.edu

Report Number:
85-556

ON A RECURRENCE EQUATION ARISING IN THE
ANALYSIS OF CONFLICT RESOLUTION ALGORITHMS

Wojciech Szpankowski

CSD-TR-556
November 1985

ON A RECURRENCE EQUATION ARISING IN THE ANALYSIS OF CONFLICT RESOLUTION ALGORITHMS

*Wojciech Szpankowski**
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907, USA

Abstract

Conflict resolution algorithms (CRA) for broadcast communications have become increasingly popular since the work of Capetanakis as well as Tsybakov and Mikhailov (CTM-algorithm). In this paper we consider a class of CTM algorithms for which a common recurrence equation for the expected length of the conflict resolution interval is found. An analysis of the equation is the primary goal of this paper. A closed form expression for the solution of the recurrence is given. Then we present an asymptotic approximation of it. In addition, we solve a functional equation associated with the recurrence and study a small value as well as an asymptotic approximation of the solution. Finally, we apply these approximations to compute maximum throughput of some CRA algorithms. We also point out that the studies are not only restricted to analysis of CRA algorithms, and a wide class of algorithms might be investigated by the proposed recurrence equation.

Categories and Subject Descriptions: C.4 **[Performance of Systems]**: Model Techniques: D.4.8 **[Operating Systems]**: Performance: *Queueing theory*; F2.2 **[Analysis of Algorithms and Problem Complexity]**: Nonnumerical algorithms and Problems: - *computations on discrete structures.*

General Terms: Performance, Theory, Algorithms

Additional Key Words and Phases: Conflict resolution algorithms, recurrence equation, asymptotic approximation, performance analysis.

1. INTRODUCTION

In a broadcast packet-switching network a finite or infinite number of users share a common communication channel. If no central coordination is provided, then packet collisions are inevitable. The problem is to find an efficient algorithm for retransmitting conflicting packets. A variety of *conflict resolution algorithms* (CRA) have become more and more popular since the work of Capetanakis [3] [4] as well as Tsybakov and Mikhailov [16] [17]. The common assumptions specifying the environment are:

- infinite number of users
- a single, error-free channel is available
- the channel time is slotted, and a slot duration is equal to a packet transmission time
- propagation delay is negligible
- the users are identical
- at the end of any slot each user can determine a status of the slot, that is, with a *binary feedback channel* a user distinguishes only between collision and no collision (*something/nothing*), and with *ternary feedback channel* a user recognizes idle, success or collision slot.

The basic idea of CRA is to solve each conflict through a *conflict resolution interval* (CRI). In such an interval a conflict of multiplicity n is partitioned into conflicts of multiplicity smaller or equal to n, and this process is continued as long as n conflicts of multiplicity one (success) are reached. The partition can be made on the basis of a random variable [3] [4], [13], [16] or on the basis of the time when the user became active [1], [6], [17]. Many modifications of the basic algorithm are possible depending on the additional information acquired during a CRI [2] [9] [10] [12] [13] (for more details see also next section).

The average and higher moments of a CRI length are an important information needed to determine the maximum throughput and to compute other issues characterizing the algorithms. It is proved that the expected CRI length satisfies a linear recurrence equation [9] ,[13] which has a common form for a class of CRA algorithms. Each algorithm in that class is modelled by this equation with an appropriate additive term. Although we restrict our considerations to CRA algorithms, there are many other algorithms in the computer science field which might be analyzed by this recurrence. Therefore, study of the recurrence equation is the primary goal of this work.

This paper is organized as follows. In the next section we describe three CRA algorithms which are considered as motivating examples for studies of a linear recurrence equation in Section 3. To cover a wide class of algorithms (in particular CRA algorithms) we assume that an additive term in the equation is any sequence of numbers. Under such an assumption we solve the recurrence and present an asymptotic approximation of it. In addition, for some cases we find a solution of the functional equation for the generating function associated with the recurrence. Then, a small value and asymptotic approximation for the generating function is presented. Finally, in Section 4 we apply the studies to throughput analysis of the three algorithms discussed in Section 2.

Previous analysis of CRA algorithms was mainly restricted either to direct numerical computations of the basic recurrence [1], [2], [12], [16], [17] or to analytical solution and asymptotic approximation of a given recurrence describing a conflict resolution algorithm [5], [9], [13]. We extend these analyses in the sense that a class of CRA algorithms is studied through a common recurrence equation.

2. CONFLICT RESOLUTION ALGORITHMS (CRA)

In this section we shortly describe three CRA algorithms, which are considered as motivating examples for general studies of Section 3. The first algorithm presented here is *static V-ary tree algorithm with arbitrary biased coins and binary feedback* [13], i.e., it is a generalization of *Capetanakis-Tsybakov-Mikhailov* (CTM) algorithm [3], [4], [16] with asymmetric tree. We call it, in short, a CTM-algorithm with V-ary asymmetric tree. Next we discuss *static modified V-ary tree algorithm with arbitrary coins and ternary feedback*, which is called here modified CTM-algorithm with V-ary asymmetric tree. For both algorithms a collision is partitioned on the basis of a random variable (stack-type algorithms) in the contrary to the third discussed algorithm where a partition of a collision is made on the basis of the time when the user became active (interval-searching algorithm [6], [17]). More precisely, the last algorithm is a *dynamic V-ary tree algorithm with multibit OR-channel overhead and binary feedback*, called here interval searching algorithm (abbreviated ISA) with multibit overhead.

2.1 CTM-algorithm with V-ary asymmetric tree

We assume a binary feedback channel and blocked-access protocol [12], [13], that is, a user recognizes only *collision/no collision*, and new packets remains blocked until the current conflict resolution interval (CRI) terminates. Then, the algorithm works as follows:

i) Whenever a user becomes active it tries to transmit a packet in the next slot. The access protocol specifies who is allowed to do so.

ii) Each active user maintains a conceptual *global stack*, and at each slot-end it specifies its position in the stack according to the following procedure:

1. All users at level 0 are allowed to transmit their packets in the nearest slot.
2. If it was not a collision slot, then a user becomes inactive and all other users decrease their stack level by 1.
3. If it was a collision slot, then all users at stack level i >= 1 change to level i + V-1. The users at level 0 are randomly split into V groups and they are placed at 0,1,. . . ,V-1 levels. The partition is made on the basis of a random variable, that is, each user at level zero selects randomly and independently of the other active users an integer in the range [0, V-1] with probabilities $p_1, p_2, \ldots, p_V$, respectively.

iii) This algorithm is repeated as long as the initial conflict is resolved. To know when the original collision is solved each user has a *counter* which is set initially to zero, incremented by V-1 for each collision and decremented by one for any non-collision slot. When the counter is decremented to -1, then the original collision is resolved, and the counter is zeroed again.

This algorithm is summarized below in the procedure RESOLVE, which is activated at each user at the end of any slot:

```
procedure RESOLVE    (V: integer; var top: integer; M: array [1..max: integer] of integer);
{M represents global stack, and M(i) contains the number of packets at level i; counter is a global variable
which indicates when a conflict is resolved; RANDOM is a procedure which returns a random number in
the range [0..V-1] with probabilities p_1, p_2, ..., p_V, respectively}

var     i,k: integer;
        collision: boolean;
begin
        if M(0) = 0 or M(0) = 1 then begin
        collision: = false; counter: = counter - 1;
        for i: = 1 to top do M(i-1):=M(i); top:=top -1
        end
else begin
        collision: = true; counter: = counter + V-1;
        for i=1 to top do M(i+V-1):=M(i); top:=top+V-1
        end
if counter = -1 then counter:=0 {conflict resolved,restart counter}
else begin
        for i = 1 to M(0) do begin
                RANDOM (k);
                M(k):=M(k) + 1
        end
end; {RESOLVE}
```

To analyze the algorithm let $\mathbf{N}_k$ denote the number of users transmitting at the first slot of the k-th CRI. We call $\mathbf{N}_k$ multiplicity of the k-th conflict, and we omit the index k when no confusion can arise. Assume now $\mathbf{N}=n$, then by $\mathbf{L}_n$ (random variable) we denote the conditional length of a CRI with multiplicity n, and by $\mathbf{I}_i$, $i = 1,2,\ldots,V, \mathbf{I}_1+\mathbf{I}_2+\ldots+\mathbf{I}_V = n$ we define the number of packets at level $i-1$ in the global stack after random distribution of the collision. Then, the algorithm implies that

$$\begin{aligned} &\mathbf{L}_n = 1 \qquad n = 0,1 \\ &\mathbf{L}_n = 1 + \mathbf{L}_{\mathbf{I}_1} + \mathbf{L}_{\mathbf{I}_2} + \ldots + \mathbf{L}_{\mathbf{I}_V} \qquad n \geq 2 \end{aligned} \tag{1}$$

and

$$Pr\{\mathbf{I}_1 = j_1, \mathbf{I}_2 = j_2, \ldots, \mathbf{I}_V = j_V \mid \mathbf{N} = n\} = \frac{n!}{j_1! j_2! \ldots j_V!} p_1^{j_1} p_2^{j_2} \ldots p_V^{j_V}, \qquad \sum_{k=1}^{V} j_k = n \tag{2}$$

Denoting

$$\begin{pmatrix} n \\ \mathbf{j} \end{pmatrix} \stackrel{def}{=} \frac{n!}{j_1! j_2! \ldots j_V!} \qquad j_1 + j_2 + \ldots j_V = n$$

and $L_n \stackrel{def}{=} E\{\mathbf{L}_\mathbf{N} \mid \mathbf{N}=n\}$, then by (1) and (2) the conditional average length of CRI, L_n, satisfies the following recurrence

$$\begin{aligned} &L_0 = L_1 = 1 \\ &L_n = 1 + \sum_{\mathbf{j}} \begin{pmatrix} n \\ \mathbf{j} \end{pmatrix} p_1^{j_1} p_2^{j_2} \ldots p_V^{j_V} (L_{j_1} + L_{j_2} + \ldots + L_{j_V}) \end{aligned} \tag{3}$$

where the sum is over all $\mathbf{j}=(j_1,\ldots,j_V)$ such that $j_1+j_2+\ldots+j_V = n$.

To determine the maximum throughput of the algorithm, $\lambda_{\max}$, note that $\mathbf{N}_k$ is a Markov chain [5], [16]. Then, by Pakes condition [14] the process $\{\mathbf{N}_k, k\geq 0\}$ is ergodic if $\limsup_{n\to\infty} E\{\mathbf{N}_{k+1} - \mathbf{N}_k \mid \mathbf{N}_k = n\} < 0$. But $E\mathbf{N}_{k+1} = \lambda E\,\mathbf{L}_{\mathbf{N}_k}$, where λ is the input rate of new packets generation. Hence, $\limsup_{n\to\infty} E\{\lambda L_n - n\} < 0$ is sufficient for ergodicity, and if $\lambda < \lambda_{\max}$ where

$$\lambda_{\max}^{-1} = \limsup_{n\to\infty} \frac{L_n}{n} \tag{4}$$

then the condition is satisfied and the algorithm is stable.

2.2 Modified CTM algorithm with V-ary asymmetric tree

The basic collision resolution mechanism is the same as described above except that ternary feedback is assumed. Note now that if after a collision $\mathbf{I}_1 = \mathbf{I}_2 = \ldots = \mathbf{I}_{V-1} = 0$, (the next $V-1$ slots are empty), then the next slot must contain a collision ($\mathbf{I}_V > 1$). This can be simply avoided if after $V-1$ consecutive empty slots following a collision slot we immediately activate procedure ii3) from the previous section. Therefore, instead of (1) we find

$$\mathbf{L}_0 = \mathbf{L}_1 = 1$$

$$\mathbf{L}_n = \begin{cases} V - 1 + \mathbf{L}_n & I_1 = I_2 = \ldots = I_{V-1} = 0 \\ 1 + \mathbf{L}_{\mathbf{I}_1} + \mathbf{L}_{\mathbf{I}_2} + \ldots + \mathbf{L}_{\mathbf{I}_V} & \textit{otherwise} \end{cases}$$

and the recurrence for L_n becomes

$$\begin{aligned} &L_0 = L_1 = 1 \\ &L_n = 1 - p_V^n + \sum_{\mathbf{j}} \binom{n}{\mathbf{j}} p_1^{j_1} \ldots p_V^{j_V} (L_{j_1} + L_{j_2} + \ldots + L_{j_V}) \quad n \geq 2 \end{aligned} \tag{5}$$

The main difference between (3) and (5) lies in the first additive term of the linear recurrences (3) and (5) for $n \geq 2$. Moreover, the same analysis as before shows that the algorithm is stable if $\lambda < \lambda_{max}$, where λ_{max} is computed as in (4).

2.3 Interval-searching algorithm with multibit overhead

In that case we assume binary feedback channel and Poisson arrival of new packets. The algorithm combines *interval-searching strategy* - introduced by Gallager [6] and Tsybakov-Mikhailov [17] - with V-ary symmetric voting mechanism employed by dedicating a small fraction of the channel capacity to *feedback overhead*. More precisely, each channel slot consists of two parts: the first one corresponds to data packet transmission , and the second part is composed of V minislots. A minislot is capable of carrying at least one bit of information. Equivalently, we may assume that a packet contains a standard data packet and V-bit subfield used for overhead purposes. By β we denote the ratio of V minislots duration and data packet transmission time.

Each time a user transmits a data packet, he also sends a pulse in one of the V minislots , and the algorithm specifies which minislot is chosen.

This is an interval-searching algorithm what means that the partition of a collision is made on the basis of time when users became active. At each step the algorithm marks a subset (an interval) of time axis called *enabled interval* (EI), and packets which fall in it are transmitted in the next slot together with pulses in appropriate minislots. The duration of the subset depends on the past outcome of the channel.

More precisely, access to the channel is controlled by a window based on the current packet age and content of minislots (something/nothing) of the current slot. Let s_i denote the starting point for i-th EI, and t_i is corresponding starting point for the conflict resolution interval (CRI). *A conflict is solved if all packets which fall into the initial EI are successfully sent in the corresponding CRI.* Initially, the enabled interval is set to be $[s_i, \min\{s_i+x, t_i\})$, where x is a constant which will be further optimized. This EI is also divided into V identical parts, say $EI_1, EI_2, \ldots, EI_V$, and packets which fall into EI_l, $l=1,2,..,V$, send a pulse in the l-th minislot . If at most one packet falls in the initial EI, then the conflict resolution interval ends immediately, and $s_{i+1}=s_i+\min\{x, t_i-s_i\}$. Otherwise, the first nonempty minislot is found, say the l^*-th, and the algorithm skip over l^*-1 parts of the EI, inspecting next the l^*-th part of the EI. The above procedure is repeated for EI_{l^*}. A CRI that begins with a collision continues until all packets from the initial EI are successfully sent. Then, the next starting point for EI is computed according to $s_{i+1}=s_i+\min\{x, t_i-s_i\}$.

This algorithm is a slight modification of multibit feedback algorithm discussed in [2] (see [2] for more detailed description of the algorithm) , however, our algorithm is FCFS (fist-come-first-serve). Moreover, contrary to the Gallager-Tsybakov-Mikhailov algorithm [6], [17] we resolve a whole initial EI before next EI is analyzed.

To investigate the algorithm let $\mathbf{N}_k$, $\mathbf{L}^k$ denote the multiplicity and the length of CRI for the k-th conflict, respectively. Let also $L_n = E\{\mathbf{L}^k \mid \mathbf{N}_k = n\}$. We normalize L_n with respect to data packet transmission time. Then, the following recurrence holds

$$L_0 = L_1 = 1+\beta$$
$$L_n = 1+\beta+\sum_{\mathbf{j}} Pr\{U_1 = j_1, U_2 = j_2, \ldots, U_V = j_V \mid \mathrm{N}=n\}\,(L^1_{j_1}+L^1_{j_2}+\ldots+L^1_{j_V})$$

where the sum is over all $\mathbf{j}$ such that $j_1+j_2+\ldots+j_V = n$, and $Pr\{U_1 = j_1, \ldots, U_V = j_V\}$ denotes the probability of $j_1, j_2, \ldots j_V$ arrivals in the first, second,. . . , $V-th$ part of a EI, while L_i^1 is the expected length of CRI for U_i-conflict. Assuming Poisson arrivals one immediately obtains (as a consequence of uniform distribution of events in a Poisson stream):

$$Pr\{U_1 = j_1, U_2 = j_2, \ldots, U_V = j_V \mid \mathrm{N}=n\} = \binom{n}{\mathbf{j}}(1/V)^n$$

Moreover, according to the algorithm rule it is clear that

$$L_j^1 = \begin{matrix} 0 & j=0 \\ L_j & j \geq 1 \end{matrix}$$

Then, after some algebra one finds

$$\begin{aligned} &L_0 = L_1 = 1+\beta \\ &L_n = (1+\beta)(1-Vq^n)+V\sum_{j=0}^{n}\binom{n}{j}p^j q^{n-j} L_j \end{aligned} \tag{6}$$

where $p = 1/V$ and $q = 1-p$.

More sophisticated analysis is necessary to determine the maximum throughput of the algorithm. It follows from the fact that $\mathbf{N}_k$ is not longer a Markov chain. Fortunately, it is proved [2], [17] that so called *transmission lag*, $\mathbf{T}_k$, defined as $\mathbf{T}_k \overset{def}{=} t_k - s_k$ is a Markov process with continuous state-space and discrete time. Then, by Tweedie's condition [18] the process is ergodic if $E\{\mathbf{T}_{k+1} - \mathbf{T}_k \mid \mathbf{T}_k \geq t\} < 0$ for $t \geq t^*$, t^* is a finite real number. But, one can show that [2], [17]

$$E\{\mathbf{T}_{k+1} - \mathbf{T}_k \mid \mathbf{T}_k > x\} = E\{\mathbf{L}^k \mid \mathbf{T}_k \geq x\} - x \tag{7}$$

On the other hand, $\{\mathbf{T}_k \geq x\}$ implies that the length of the k-th enabled interval EI is equal to x, so the average number of arrivals in this interval, μ, is equal to $\mu = \lambda x$, where λ is the arrival rate for the Poisson process. By (7) the following condition $x > E\{\mathbf{L}^k |\ \mathbf{T}_k = x\} = E\{\mathbf{L} | EI = x\}$ is sufficient for ergodicity and the maximum throughput of the algorithm is

$$\lambda_{\max} = \sup_{\mu} \frac{\mu}{F(\mu)} \quad , \quad F(\mu) = \sum_{n=0}^{\infty} L_n \frac{\mu^n}{n!} e^{-\mu} \tag{8}$$

Note that $F(\mu)e^{\mu}$ is exponential generating function for L_n.

3. A RECURRENCE EQUATION AND SOME APPROXIMATIONS

Generalizing the above three examples we consider a sequence L_n, $n = 0,1,...$ which satisfies the following recurrence

$$\begin{aligned} &L_0, L_1 - \text{given} \\ &L_n = a_n + b \sum_{\mathbf{j}} \frac{n!}{j_1! j_2! ... j_V!} p_1^{j_1} p_2^{j_2} ... p_V^{j_V} (L_{j_1} + L_{j_2} + ... + L_{j_m}) \end{aligned} \tag{9}$$

where the sum is over all $\mathbf{j}$ such that $j_1 + j_2 + ... + j_V = n$, and V, b, $m \le V$ are constants, a_n is a given sequence, and $\sum_{i=1}^{V} p_i = 1$. Let

$$L(z) = \sum_{n=0}^{\infty} L_n \frac{z^n}{n!}$$

be exponential generating function for L_n. Then, after some algebra we find the following functional equations for $L(z)$

$$L(z) - b \sum_{i=1}^{m} L(p_i z) \exp[(1-p_i)z] = A(z) - l_0 - l_1 z \tag{10}$$

where $A(z)$ is exponential generating function for a_n, and

$$\begin{aligned} &l_0 = a_0 + L_0(mb - 1) \\ &l_1 = a_1 + b(L_1 - L_0) \sum_{i=1}^{m} p_i + L_0 mb - L_1 \end{aligned} \tag{11}$$

Let us define now a new function $H(z) = L(z)e^{-z}$. Multiplying both sides of (10) by e^{-z} we find

$$H(z) - b \sum_{i=1}^{m} H(p_i z) = A(z)e^{-z} - l_0 e^{-z} - l_1 z e^{-z} \tag{12}$$

This functional equation is suitable for establishing a closed form expression for L_n. Therefore, let us introduce a sequence $\hat{a}_n$ defined as

$$\hat{a}_n = \sum_{k=0}^{n} \begin{pmatrix} n \\ k \end{pmatrix} (-1)^k a_k \tag{13}$$

Note also that

$$A(z)e^{-z} = \sum_{n=0}^{\infty} \hat{a}_n (-1)^n \frac{z^n}{n!} = \hat{A}(-z) \tag{14}$$

Relationship (13) is well known in the combinatorial analysis. In fact, a_n and $\hat{a}_n$ represent so called *inverse relations* [10] ,[15] that is, $\hat{\hat{a}}_n = a_n$. In [15] a number of inverse relations are presented. In particular,

$$a_n = \begin{pmatrix} n \\ r \end{pmatrix} c^n \quad \hat{a}_n = \begin{pmatrix} n \\ r \end{pmatrix} (-c)^r (1-c)^{n-r} \tag{15}$$

where δ_{nm} is Kronecker delta.

Now a closed form solution for L_n might be established. Define h_n as a coefficient in Taylor expansion of $H(z)$, that is, $H(z) = \sum_{n=0}^{\infty} h_n z^n$. Let also k^* be such an index k that

$$1 - b \sum_{i=1}^{m} p_i^{k^*} = 0 \tag{16}$$

Then, equating the coefficients of power z in (12), and taking into account (14), (16) we find

$$h_k = \begin{cases} l_0 & k = 0 \\ \dfrac{kl_1 + \hat{a}_k - l_0}{k!(1 - b \sum_{i=1}^{m} p_i^k)} & k > 0 \text{ and } k \neq k^* \end{cases}$$

However, the relation $L(z) = H(z)e^{-z}$ implies that

$$L_n = n! \sum_{k=0}^{n} \frac{h_k}{(n-k)!}$$

Moreover, h_{k^*} may be found either from the boundary conditions (L_0, L_1) or from recurrence equation itself. For example, we may apply formula

$$h_{k^*} = \frac{L_{k^*}}{k^*!} - \sum_{i=0}^{k^*-1} \frac{h_i}{(k^*-i)!} \tag{17}$$

and for $b=1$ (17) becomes $h_1 = (L_1 - L_0)$, where $k^*=1$. Finally,

$$L_n = L_0 + \frac{n!}{(n-k^*)!} h_{k^*} + \sum_{\substack{k=1 \\ k \neq k^*}}^{n} (-1)^k \binom{n}{k} \frac{kl_1 + \hat{a}_k - l_0}{1 - b \sum_{i=1}^{m} p_i^k} \tag{18}$$

In some cases we can optimize L_n with respect to $\mathbf{p}$. For example, if $L_0 = L_1$, $m=V$, $b=1$, and $\hat{a}_k$ does not depend on $\mathbf{p}$, then it is easy to prove that

$$L_n = \min \quad \text{for all} \quad n \quad \text{iff} \quad p_i = 1/V, \; i = 1,2,...V \tag{19}$$

what suggests that V-ary symmetric tree is optimal in this case.

Asymptotic approximation

It should be noted that the equation for L_n given by (18) is neither useful for direct computations of L_n nor interesting for throughput analysis. It is a consequence of the fact that the factor $(-1)^n$ leads to numerical instabilities for $n > 20$, and the formula is too complex to derive some qualitative properties of L_n. However, (18) might be used to establish an asymptotic approximation for L_n and produce easily verifiable conditions for stability of algorithms (at least for the first two algorithms discussed in the previous section). Therefore, we now deal with asymptotic analysis of L_n for $n \to \infty$. Naturally, the most difficult to handle is the sum in (18). It is not reasonable to derive an asymptotic approximation for any sequence a_n, therefore, we restrict a class of the sequences to $a_n = \binom{n}{r} c^n$, where c is a constant and r is an integer. Then, $\hat{a}_n$ is given by (15). Note also that the first and the third term in the numerator of (18) may be considered as special cases of the sequence a_n, namely for $r=1, c=1$ and $r=0, c=1$, respectively. Concluding, asymptotic analysis of (18) with a_n given by (15) might be easily derived from asymptotic analysis of the following

$$S(n,r,\mathrm{d},c) = \sum_{k=2}^{n} (-1)^k \binom{n}{k} \binom{k}{r} \frac{c^k}{D - \sum_{i=1}^{m} d_i^k} \tag{20}$$

where $\sum_{i=1}^{m} d_i = D$, $0 \le d_i < 1$, c is a constant and r is a non-negative integer. We often write

$S(n,r)$ instead of $S(n,r,\mathbf{d},c)$.

In the further part of this section we focus our attention on (20). For simplicity of the following derivations we consider separately three cases, namely $r=0$, $r=1$ and $r\geq 2$.

Case $r=0$. Expanding the reciprocal of the denominator of (20) in a geometric series we find that

$$S(n,0)=\sum_{k=2}^{n}(-1)^k \binom{n}{k} \frac{c^k}{D-\sum_{i=1}^{m} d_i^k} = D^{-1}\sum_{k=2}^{n}(-1)^k \binom{n}{k} c^k \sum_{l=0}^{\infty} D^{-l} (\sum_{i=1}^{m} d_i^k)^l =$$

$$D^{-1}\sum_{k=2}^{n}(-1)^k \binom{n}{k} \sum_{l=0}^{\infty} D^{-l} \sum_{\mathbf{j}} \binom{l}{\mathbf{j}} (c \prod_{s=1}^{m} d_s^{j_s})^k \tag{21}$$

where the last sum is over all $\mathbf{j}$ such that $j_1+j_2+\ldots+j_m=l$. Let us define now $\phi = c\prod_{S=1}^{n} d_s^{j_i}$.

Note also that [15]

$$\sum_{k=2}^{n}(-1)^k \binom{n}{k} \phi^k = (1-\phi)^n + n\phi - 1$$

hence (21) becomes

$$S(n,0)=D^{-1}\sum_{l=0}^{\infty} D^{-1} \sum_{\mathbf{j}} \binom{l}{\mathbf{j}} [(1-\phi)^n + n\phi - 1]$$

Introducing $x=n\phi$ and noting that $(1-\frac{x}{n})^n = e^{-x}+x^2 O(n^{-1})$ we may approximate $S(n,0)$ by

$$T(n,0)=D^{-1}\sum_{l=0}^{\infty} D^{-1} \sum_{\mathbf{j}} \binom{l}{\mathbf{j}} [e^{-x}+x-1]$$

But using Mellin transform we find that for $x>0$ [8],[11]

$$e^{-x}+x-1=\int_{(-3/2)} \Gamma(z) x^{-z} dz \tag{22}$$

where $\Gamma(z)$ is gamma function [8] and the notation $\int_{(c)}$ stands for $\frac{1}{2\pi i}\int_{c-i\infty}^{c+i\infty}$. Then, for $\mathbf{Re}(z)<0$

$$T(n,0)=D^{-1}\int_{(-3/2)} \Gamma(z) \sum_{l=0}^{\infty} D^{-l} \sum_{\mathbf{j}} \binom{l}{\mathbf{j}} n^{-z}\phi^{-z} = \int_{(-3/2)} \frac{\Gamma(z) n^{-z} c^{-z} dz}{D-\sum_{i=1}^{m} d_i^{-z}} \tag{23}$$

since an appropriate geometric series is convergent for $\mathbf{Re}(z)<0$.

Case $r=1$. Let us compute $S(n+1,1)$ instead of $S(n,1)$. Then, by the same arguments as above we obtain

$$S(n+1,1)=D^{-1}\sum_{l=0}^{\infty}D^{-l}\sum_{\mathbf{j}}\begin{bmatrix} l \\ \mathbf{j}\end{bmatrix}\sum_{k=2}^{n+1}(-1)^k\begin{bmatrix} n+1 \\ k\end{bmatrix}k\phi^k \tag{24}$$

Noting that [15]

$$\sum_{k=2}^{n+1}(-1)^k\begin{bmatrix} n+1 \\ k\end{bmatrix}k\phi^k=(n+1)\phi[1-(1-\phi)^n]$$

and evaluating the expression in the square brackets of the above by $1-e^{-x}$, where $x=n\phi$, we finally approximate $S(n+1,1)$ by $T(n+1,1)$, where

$$T(n+1,1)=D^{-1}\sum_{l=0}^{\infty}D^{-l}\sum_{\mathbf{j}}\begin{bmatrix} l \\ \mathbf{j}\end{bmatrix}\phi[1-e^{-x}](n+1)$$

But , by Mellin transform [8], [11]

$$1-e^{-x}=-\int_{(-1/2)}\Gamma(z)x^{-z}dz \tag{25}$$

hence, manipulating terms of (24) we obtain for $\mathbf{Re}(z)<1$

$$T(n+1,1)=-(n+1)\int_{(-1/2)}\frac{\Gamma(z)n^{-z}c^{1-z}dz}{D-\sum_{i=1}^{m}d_i^{1-z}} \tag{26}$$

where an appropriate series in (26) is convergent for $\mathbf{Re}(z)<1$.

Case $r\geq 2$. We compute $S(n+r,r)$ instead of $S(n,r)$. The same procedure as above applied to $S(n,r)$ gives formula similar to (24), however, the last sum becomes

$$\sum_{k=2}^{m+r}(-1)^k\begin{bmatrix} m+r \\ k\end{bmatrix}\begin{bmatrix} k \\ r\end{bmatrix}\phi^k=\sum_{k=2}^{m+r}(-1)^k\begin{bmatrix} m+r \\ r\end{bmatrix}\begin{bmatrix} n \\ k-r\end{bmatrix}\phi^k=(-1)^r\begin{bmatrix} m+r \\ r\end{bmatrix}\phi^r(1-\phi)^n$$

Approximating $(1-\phi)^n=(1-x/n)^n\ \ e^{-x}+x^2O(n^{-1})$ and noting that

$$e^{-x}=\int_{(1/2)}\Gamma(z)x^{-z}dz \tag{27}$$

we finally find an approximation $T(n+r,r)$ of $S(n+r,r)$ where

$$T(n+r,r)=(-1)^r\begin{bmatrix} n+r \\ r\end{bmatrix}\int_{(1/2)}\frac{\Gamma(z)n^{-z}c^{r-z}dz}{D-\sum_{i=1}^{m}d_i^{r-z}} \tag{28)}$$

assuming $\mathbf{Re}(z) < r$.

□

To justify the above approximations we prove that

Theorem 1. For all r and n

$$S(n,r) = T(n,r) + O(1)$$

Proof. The proof is based on the idea presented in [9]. We assume that $r \geq 2$ (for r=0,1 the proof is similar). Let us denote $\delta(\mathbf{d},n) = T(n+r,r) - S(n+r,r)$. Then

$$\delta(\mathbf{d},n) = (-1)^r \binom{n+r}{r} D^{-1} \sum_{l=0}^{\infty} D^{-l} \sum_{\mathbf{j}} \binom{l}{\mathbf{j}} \phi^r \left[e^{-x} - (1-x/n)^n \right] \tag{29}$$

Note that

$$e^{-x} - (1-x/n)^n \leq \frac{x^2}{n} e^{-x} < \frac{x}{n} \tag{30}$$

where the last inequality follows from $xe^{-x} < 1, x>0$. Now split the sum over l in (29) into three parts $0 \leq l \leq n_1, n_1 \leq l < n_2, l \geq n_2$, where

$$n_1 = [\ln n - \ln(r+1)\ln n]/\ln d_s^{-1} \quad n_2 = \ln n / ln(D/a)$$

and

$$d_s = \min_{1 \leq i < V} d_i, \quad a = \sum_{i=1}^{m} d_i^{r+1}$$

For $0 \leq l < n_1$, we use the first inequalities of (30). Then

$$\delta_1(\mathbf{d},n) = O\left(n^{r+1} D^{-1} \sum_{l=0}^{n_1-1} D^{-l} \sum_{\mathbf{j}} \binom{l}{\mathbf{j}} \phi(\mathbf{j})^{r+1} e^{n\phi(\mathbf{j})} \right)$$

where in the last expression we show explicit that ϕ depends on $\mathbf{j}$. But, by the above definitions of d_s and $\phi(\mathbf{j})$, we find that $\phi(\mathbf{j}) \geq d_s^l > d_s^{n_1} = \phi^*(n_1)$ and $e^{-n\phi(\mathbf{j})} \leq e^{-n\phi^*(n_1)}$. Therefore,

$$\delta_1(\mathbf{d},n) < O\left(n^{r+1} e^{-n\phi^*(n_1)} D^{-1} \sum_{l=0}^{n_1-1} D^{-l} \sum_{\mathbf{j}} \binom{l}{\mathbf{j}} \phi^{r+1}(\mathbf{j}) \right) = O(\exp[(r+1)\ln n - n\phi^*(n_1)]$$

Note that by our choice of n_1 one finds that $(r+1)\ln n - n\phi^*(n_1) < 0$, so $\delta_1(\mathbf{d},n) < O(1)$.

For $l > n_2$ we use the second inequality in (30). Then

$$\delta_3(\mathbf{d},n) = O\left[n^r \sum_{l \ge n_2} D^{-l-1} \sum_{\mathbf{j}} \begin{pmatrix} l \\ \mathbf{j} \end{pmatrix} \phi(j)^{r+1}\right] < O\left[n^r \sum_{l \ge n_2} D^{-l-1} [\sum_{i=1}^{m} d_i^{r+1}]^l\right] =$$

$$O(n^r D^{-n_2} a^{n_2}/(D-a)) = O(n^r (a/D)^{n_2}) = O(\exp[r \ln n + n_2 ln(a/D)])$$

and under our choice of n_2 the exponent is negative, so $\delta_3(\mathbf{d},n)$ is $O(1)$.

Finally, for $n_1 \le l < n_2$ we apply discrete version of mean value theorem and arguing as in [9] we prove that $\delta_2(\mathbf{d},n) < O(1)$.

□

By Theorem 1 the evaluation of $S(n,r)$ is reduced to computation of $T(n,r)$. A suitable formula for $T(n,r)$ is given in the following

Corollary 1. For any n , and r

$$T(n+r,r) = (-1)^r \frac{(n+r)c}{r!}[1 + O(n^{-1}]G(n,r) \tag{31a}$$

where

$$G(n,r) = \int_{(1/2-[2-r]^+)} \frac{\Gamma(z)(nc)^{r-1-z} dz}{D - \sum_{i=1}^{m} d_i^{r-z}} \tag{31b}$$

and $a^+ = \max\{a, 0\}$.

Proof.. By (23), (26) and (28) we find immediately that

$$T(n+r,r) = (-1)^r \begin{pmatrix} n+r \\ r \end{pmatrix} \int_{(1/2-[2-r]^+)} \frac{\Gamma(z) n^{-z} c^{r-z} dz}{D - \sum_{i=1}^{m} d_i^{r-z}}$$

Noting now that

$$\begin{pmatrix} n+r \\ r \end{pmatrix} \frac{n}{n^r} = (n+r)\frac{1}{r!}[1+1/n][1+2/n] \cdots [1+(r-1)/n] = \frac{n+r}{r!}[1+O(n^{-1})]$$

we obtain (31).

□

The evaluation of the contour integral in (31) is routine: one goes from $(a, -iN_1)$ to (a, iN_1) to (N_2, iN_1) to $(N_2 - iN_1)$ to $(a, -iN_1)$ in a negative sense, where $a = ½ - [2-r]^+$, $r \ge 0$. For $N_1 \to \infty$ the horizontal parts of the integral vanish since

$|\Gamma(t+iN)| = O(|t+iN|^{t-½} e^{-t-\pi N/2})$ [19], while vertical component $(N_2, -iN_1)$ decays due to the factor n^{r-1-z} [8], [19]. Hence, the required integral is minus the sum of residues of the function under the integral to the right of the vertical line fixed at point $a = ½ - [2-r]^+$. The function under integral is analytical except the roots of the denominator and eventually poles of the gamma function. Let us denote by z_k^r, $r = 0,1,...,$ $k = 0,\pm1,...$ the roots of the denominator, that is

$$D - \sum_{i=1}^{m} d_i^{r-z} = 0 \tag{32}$$

This equation is quite difficult to analyze, and only for few isolated values of d may be solved [5],[11],[13] , and it is proved that the roots of (32) are well separated [5]. Naturally, for k=0 we have $z_0^r = r-1$. Note also that $z_0^0 = -1$ and $z_0^1 = 0$ coincide with the poles -1, 0 of the gamma function, while for $r \geq 2$ $z_k^r \neq$ *nonpositive integer.* Let us denote by $g_r(z)$ the function under the integral. Then

$$G(n,r) = \int_{(1/2-[2-r]^+)} g_r(z)dz = -[1-r]^+ \operatorname{res} g_0(0) - \operatorname{res} g_r(r-1) - \sum_{z_k^r \neq 0, r-1} \operatorname{res} g_r(z_k^r) \tag{33}$$

where $\operatorname{res} g(\xi) = \operatorname*{res}_{z=\xi} g(z)$. Let us also denote the sum in (33) as $f_{r-1}(nc)$. Then

$$f_{r-1}(n) = -\sum_{z_k^r \neq 0, r-1} \frac{\Gamma(z_k^r) n^{r-1-z_k^r} dz}{\sum_{i=1}^{m} d_i^{r-z_k} \ln d_i} \tag{34}$$

or in another form

$$f_{r-1}(n) = \int_{(1/2)} \frac{\Gamma(z) n^{r-1-z} dz}{D - \sum_{i=1}^{m} d_i^{r-z}} + \frac{\Gamma(r-1)}{\sum_{i=1}^{m} d_i \ln d_i} \tag{35}$$

This sum is quite difficult to evaluate. In particular, it turns out [10], [11], [13] that the function $f_{r-1}(n)$ does not necessary have a limit as $n \to \infty$ (fluctuating function). Fortunately, for small value of V $f_{r-1}(n)$ is extremely small and may be safely ignored in practical calculations (see also Section 4).

The second term in (33) is also easy to evaluate for $r > 1$, since

$$res\, g_r(r-1) = \frac{\Gamma(r-1)}{\sum_{i=1}^{m} d_i \ln d_i} \quad r > 1 \tag{36}$$

however, for $r = 0,1$ more sophisticated computations are needed to compute it. In Appendix we prove that

$$\mathrm{res}\, g_0(0) = \frac{1}{nc\,[D-m\,]} \tag{37a}$$

$$\mathrm{res}\, g_0(-1) = -\frac{\ln(nc) + \gamma - 1}{h_{\mathbf{d}}} - \frac{h_{\mathbf{d}}^{(2)}}{2h_{\mathbf{d}}^2} \tag{37b}$$

$$\mathrm{res}\, g_1(0) = +\frac{\ln(nc) + \gamma}{h_{\mathbf{d}}} + \frac{h_{\mathbf{d}}^{(2)}}{2h_{\mathbf{d}}^2} \tag{37c}$$

where γ=0.57721 is the Euler constant, and

$$h_{\mathbf{d}} = -\sum_{i=1}^{m} d_i \ln d_i \;, \quad h_{\mathbf{d}}^{(2)} = \sum_{i=1}^{m} d_i (\ln d_i)^2$$

Thus we have shown that

Proposition 1. For any n and r the following holds

$$S(n,0) = nc\{\frac{\ln(nc) + \gamma - 1}{h_{\mathbf{d}}} + \frac{h_{\mathbf{d}}^{(2)}}{2h_{\mathbf{d}}^2} + f_{-1}(nc)\} - \frac{1}{D-m} + O(1)$$

$$S(n,1) = nc\{\frac{\ln(n-1)c + \gamma}{h_{\mathbf{d}}} + \frac{h_p^{(2)}}{2h_{\mathbf{d}}^2} - f_0((n-1)c)\} + O(1) \tag{38}$$

$$S(n,r) = (-1)^r \frac{nc}{r!}\{\frac{(r-2)!}{h_{\mathbf{d}}} + f_{r-1}((n-r)c)\} + O(1)$$

Proof.. It follows immediately from Theorem 1, Corollary 1 and the above analysis.

□

Formulas (38) might be used to evaluate an asymptotic approximation for

$$S(n,P_N) = \sum_{k=2}^{n} (-1)^k \binom{n}{k} \frac{P_N(k)c^k}{D - \sum_{i=1}^{m} d_i^k}$$

where $P_N(k)$ is a polynomial of k, i.e.

$$P_N(k) = \sum_{i=0}^{N} e_s k^s \;,$$

where e_i are given coefficients. To show it, note that [11]

$$k^s = \sum_{j=0}^{s} \left\{ {s \atop j} \right\} \binom{k}{j} j!$$

where $\left\{ \begin{matrix} s \\ j \end{matrix} \right\}$ is a Stirling number of second kind. Then,

$$S(n,P_N)=\sum_{s=0}^{N} e_s \sum_{r=0}^{s} \left\{ \begin{matrix} s \\ r \end{matrix} \right\} r!\, S(n,r)$$

where $S(n,r)$ is given in (38).

Functional equation

To compute maximum throughput for ISA with multibit overhead we must evaluate the function $F(\mu)$ which is defined in (8). But $F(\mu)=L(\mu)e^{-\mu}=H(\mu)$ where $L(\mu)$ is exponential generating function of L_n, and $H(\mu)$ satisfies functional equation (12). Approximations of $L(z)$ and $H(z)$ might be computed from (38), but a direct solution of (12) gives better insight into the behavior of the algorithms and produces better approximation. In this subsection, we solve this function equation for a special case which will be further used to evaluate λ_{max} for the third algorithm. Note also that $H(\mu)$ is unconditional average length of CRI. Moreover, many other quantities of interest might be calculated through $H(\mu)$. Therefore, an explicit or approximate formula for $H(\mu)$ is very important for detailed analysis of the algorithm.

Solution of functional (12) is too troublesome in its present form. Therefore, for simplicity we solve it only for two cases: either we assume $m=1$ or $p_1=p$ or $p_1=p_2=\cdots=p_m=p$. Moreover, instead of finding $H(z)$ we put $h(z)=H(z)-L_0$ (note that now $h(0)=0$) and (12) under the above assumption is

$$h(z)=\alpha h(zp)+f(z)e^{-z} \tag{39}$$

where

$$f(z)=A(z)-l_0-l_1 z+L_0 e^{z}(\alpha-1) \tag{40a}$$

$$l_0=a_0+L_0(\alpha-1) \tag{40b}$$

$$l_1=a_1+\alpha p(L_1-L_0)+L_0\alpha-L_1 \tag{40c}$$

$$\alpha = \begin{cases} b & \text{for } m=1, p_1=p \\ b\,m & \text{for } p_1=p_2= \quad \cdots \quad =p_m=p \end{cases} \tag{40d}$$

Iterating (3a) n-times and letting $n \to \infty$ we find that

$$h(z)=h^*(z)+\sum_{k=0}^{\infty} \alpha^k f(p^k z)e^{-zp^k} \tag{41}$$

where $h^*(z)=\lim_{k\to\infty} \alpha^{k+1} h(p^{k+1}z)$. Eq.(41) holds if $h^*(z)$ exists and the series is convergent. By D'Alembert's criterion [8] the latter is satisfied if $\alpha \lim_{k\to\infty} f(p^{k+1}z)/f(p^k z) < 1$. But $h(0)=0$ implies $f(0)=0$, hence l'Hospital rule shows that the following condition

$$\alpha p \lim_{k\to\infty} \frac{f'(p^{k+1}z)}{f'(p^k z)} < 1 \tag{42}$$

is sufficient for the series in (41) to be convergent . If $f'(0)\neq 0$ then $\alpha p < 1$ implies convergence in (41). For $f'(0)=0$ by l'Hospital rule we show that

$$\alpha p^2 \lim_{k\to\infty} \frac{f''(p^{k+1}z)}{f''(p^k z)} < 1 \tag{43}$$

must be satisfied. We prove that

Corollary 2. If $p\alpha=1$ $(0<p<1)$ and $f''(0)\neq 0$, then (41) is a solution of (39)-(40) for any z with $h^*(z)=z(L_1-L_0)$.

Proof. By (40a) one shows that $f(0)=0$ and $f'(0)=0$ for $p\alpha=1$. Hence by (43) the series is convergent if $p<1$ what is assumed. Moreover, if the series is convergent then $\lim_{k\to\infty} \alpha^k f(p^k z)e^{-zp^k}=0$ must be satisfied. Since $\alpha=1/p>1$ and $f(0)=0$ this is equivalent to $(u=p^k z)$

$$\lim_{k\to\infty} \alpha^k f(p^k z) = \lim_{k\to\infty} p^{-k} f(p^k z) = z \lim_{u\to 0} \frac{f(u)}{u} = z \lim_{u\to 0} f'(0) = 0$$

The last equality holds since $f'(0)=0$. To prove the formula for $h^*(z)$ note that

$$h^*(z) = \lim_{k\to\infty} \alpha^{k+1} h(p^{k+1}z) = z \lim_{u\to 0} \frac{h(u)}{u} = z\, h'(0)$$

and finding $h'(0) = L_1 - L_0$ one proves the corollary.

□

In Corollary 2 we have restricted our analysis to $p\alpha = 1$ since it is the most interesting case for us. However, the same idea might be used for $p\alpha \neq 1$.

Formula (41) is not very useful for computation, and - what is most important - it is not suitable for some approximations. Therefore, we prove

Theorem 2. Under assumptions of Corollary 2 the following holds

$$H(z) = L_0 + z(L_1 - L_0) + \sum_{k=2}^{\infty} (-1)^k \frac{\hat{f}_k z^k}{k!(1-p^{k-1})} \tag{44}$$

where

$$\hat{f}_k = \hat{a}_k - l_0 + kl_1 + L_0(\alpha - 1)\delta_{n0} \tag{45}$$

Proof. Note that by (14) $f(zp^k)e^{-zp^k} = \hat{f}(-zp^k)$. Hence the series in (41) with $\alpha p = 1$ is equal to

$$\sum_{k=0}^{\infty} p^{-k} \hat{f}(-zp^k) = \sum_{k=0}^{\infty} p^{-k} \sum_{n=2}^{\infty} (-1)^n \hat{f}_n \frac{z^n}{n!} p^{kn} = \sum_{n=2}^{\infty} (-1)^n \hat{f}_n \frac{z^n}{n!(1-p^{n-1})}$$

where the sum starts with $n = 2$ since $\hat{f}_0 = \hat{f}_1 = 0$. Formula (45) follows directly from $\hat{f}(-z) = f(z)e^{-z}$ and (40a).

□

Equation (44) is very useful for small value approximations, that is, for approximation of $H(z)$ for $z < \varepsilon$, ε is small real number. Then

$$H(z) = L_0 + z(L_1 - L_0) + \sum_{k=2}^{M} (-1)^k \frac{\hat{f}_k z^k}{k!(1-p^{k-1})} + O(z^{M+1}) \tag{46}$$

where $M > 2$, and M is rather small integer. However, for an asymptotic approximation for $H(z)$

we need a little more sophisticated analysis.

For the purpose of asymptotic analysis we use (41) and assume that a_n is given by (15), that is $A(z)=(zc)^r/r!e^{zc}$. Note also that (40a) may be rewritten in a form

$$f(z)=A(z)-a_0 a_1 z+L_0(\alpha-1)[e^z-z-1]$$

Introducing

$$F_{r,c}\overset{def}{=}\frac{(zc)^r}{r!}e^{zc}-a_0-a_1 z$$

we find that

$$f(z)=F_{r,c}(z)+L_0(\alpha-1)F_{0,1}(z) \tag{47}$$

and analysis of (41) is reduced to an asymptotic approximation of the following series

$$s(z,r,c)=\sum_{k=0}^{\infty}p^{-k}F_{r,c}(zp^k)e^{-zp^k} \tag{48}$$

Note that $a_0=a_1=0$ for $r\geq 2$, $a_0=0$ for $r=1$ and $a_0\neq 0$ for $r=0$. Therefore, three cases must be considered. We present below detailed analysis for $r\geq 2$, while for $r=0$, 1 only some hints and final results will be given.

Assume $r=2$. Then $a_0=a_1=0$ and one finds

$$s(z,r,c)=\frac{(zc)^r}{r!}\sum_{k=0}^{\infty}p^{k(r-1)}e^{-zp^k(1-c)}$$

For $c=1$ we find immediately

$$s(z,r,1)=\frac{z^r c^r}{r!(1-p^{r-1})} \tag{49}$$

For $0<c<1$ we use Mellin transform (27) and after some algebra we find

$$s(z,r,c)=\frac{zpc^r}{r!(1-c)^{r-1}}\int_{(1/2)}\frac{\Gamma(x)[z(1-c)]^{r-1-x}}{p-p^{r-x}}dx \tag{50}$$

But the integral is equal to $G(z,r,p,1-c)$ defined in (31b) and analyzed by (33). Hence we immediately find that

$$s(z,r,c)=\frac{zc^r}{(1-c)^{r-1}r!}\{\frac{V(r-2)!}{\ln V}+f_{r-1}[(z-r)(1-c)]\}+O(1)$$

for $0<c<1$.

For $r=1$ we have $a_1=zc$ and $a_0=0$. Then (48) becomes after some algebra

$$s(z,1,c)=zc\sum_{k=0}^{\infty}[(1-e^{-zp^k})-(1-e^{-zp^k(1-c)})]$$

Using now Mellin transform as given in (25) we find for $0<c<1$

$$s(1,c)=-zcp\{\int_{(-1/2)}\frac{\Gamma(x)z^{-x}}{p-p^{1-x}}dx-\int_{(-1/2)}\frac{\Gamma(x)[z(1-c)]^{-x}}{p-p^{1-x}}dx\} \tag{52}$$

and for $c=1$ the second term is (52) should be dropped. Thus the problem is reduced to evaluation of $G(z,1,p,1)$ and $G(z,1,p,1-c)$ (see Eq. (31b)) as it was done before.

Finally, for $r=0$ $a_0=1$ $a_1=zc$ (48) may be transformed into

$$s(z,0,c)=\sum_{k=0}^{\infty}p^{-k}\{(1-\delta_{c1})[e^{-zp^k(1-c)}-1+(1-c)p^kz]-[e^{-zp^k}-1+p^kz]-cp^kz[e^{-zp^k}-1]\}$$

Hence, using Mellin transforms (22) and (25) we may argue as before.

The following proposition summarizes the results

Proposition 2. For $z\to\infty$ and

(i) $c=1$

$$s(z,0,1)=\frac{z}{\ln V}-zp[f_0(z-1)+f_{-1}(z)]+O(1) \tag{53a}$$

$$s(z,1,1)=z\{\frac{\ln(z-1)+\gamma}{\ln V}+\frac{1}{2}-\frac{1}{V}f_0(z-1)\}+O(1) \tag{53b}$$

$$s(z,r,1)=\frac{z^r}{r!(1-p^{r-1})},\quad r\geq 2 \tag{53c}$$

(ii) $0 < c < 1$

$$s(z,0,c) = \frac{z}{\ln V}\{c\ln(1-\tfrac{1}{z}) + (1-c)\ln(1-c) + c\} + \tag{54a}$$
$$\frac{z}{V}\{(1-c)f_{-1}[(1-c)z] - f_{-1}(z) - cf_0(z)\} - \frac{c}{V-1} + O(1)$$

$$s(z,1,c) = zc\{\frac{\ln(1-c)^{-1}}{\ln V} - \frac{1}{V}[f_0(z-1) - f_0((z-1)(1-c))]\} + O(1) \tag{54b}$$

$$s(z,r,c) = \frac{zc^r}{(1-c)^{r-1}r!}\{\frac{V(r-2)!}{\ln V} + f_{r-1}[(z-r)(1-c)]\} + O(1) \tag{54c}$$

$\square$

Finally, by (47) and (41) we obtain

$$H(z) = L_0 + s(z,r,c) + L_0(\alpha-1)s(z,0,1) + O(1) \tag{55}$$

where $s(z,r,c)$ and $s(z,0,1)$ are computed according to (53) and (54). Summarizing, we have obtained three formulas which might be used to evaluate $H(z)$: for small values of z (46) is the most appropriate, for large values of z (53) (54) give good approximations, and for other values of z we must use (44).

4. APPLICATIONS

In this section we apply formulas (18), (38) and (53), (54) to approximate the average length of CRI for the previously described algorithms. In addition, we find maximum throughput for stable CRA algorithms, and we solve some optimization problems .

4.1 CTM algorithm

For, by (3) and (18), with $L_0 = L_1 = 1$, $b=1$, $m=V$, $a_n = 1$, $\hat{a}_n = \delta_{n0}$ we find that

$$L_n = 1 + V\sum_{k=2}^{n}(-1)^k \binom{n}{k} \frac{(k-1)}{1 - \sum_{i=1}^{V} p_i^k}.$$

Hence, $L_n = 1 + V[S(n,1,\mathbf{p},1) - S(n,0,\mathbf{p},1)]$, and by (38)

$$L_n = nV\left\{ \left[\sum_{i=1}^{V} p_i \ln p_i\right]^{-1} + f_{-1}(n) - f_0(n-1) \right\} - \frac{1}{V-1} + O(1)$$

Then maximum throughput $\lambda_{max}(\mathbf{p},V)$ is given by (4) and together with the above we obtain

$$\lambda_{max}^{-1}(\mathbf{p},V) = \lim_{n\to\infty} \sup \frac{L_n}{n} = \frac{V}{-\sum_{i=1}^{V} p_i \ln p_i} + r_1 \tag{56}$$

where $r_1 = \limsup_{n\to\infty} [f_{-1}(n) - f_0(n-1)]$. The value of r_1 is rather small compared with the leading factor in (56), hence in many computations it may be omitted, however, it depends on particular values of $p_1,...,p_V$ (see [13] and Section 4.3). The advantage of such an approximation lies also in the fact that it gives an insight into the behavior of the algorithm. Moreover, the approximation is acceptable - even if value of r_1 is relatively large - from the qualitative point of view when structured properties of algorithms are studied instead of numerical values of functions describing the algorithms (quantitative analysis). Therefore, considering now only leading factor in (56) we might easily maximized $\lambda_{max}(\mathbf{p},V)$ over p and V. In (19) we have proved that L_n is minimized for all n if and only if $p_i = 1/V$, $i = 1,2,...,V$. Then

$$\lambda_{max}(V) = \max_{\mathbf{p}} \lambda_{max}(V) = \frac{\ln V}{V}$$

and maximizing over V one finds that for $V^* = e$

$$\lambda_{max} = \max_{V} \lambda_{max}(V) = \frac{1}{e} = 0.3618,$$

but $\lambda_{max}(2) = 0.34657$ and $\lambda_{max}(3) = 0.36620$. Taking into account r_1 one finds that $\lambda_{max}(3) = 0.36611$ which is very closed to the value obtained by approximate formula.

4.2 Modified CTM algorithm

In that case (5) shows that $L_0 = L_1 = 1$ $a_n = 1 - p_V^n$. Hence, by (15) $\hat{a}_n = -(1-p_V)^n$ and (18) gives

$$L_n = 1 + \sum_{k=2}^{n} (-1)^k \frac{k(V-p_V) - (1-p_V)^k - (V-1)}{1 - \sum_{i=1}^{V} p_i^k}$$

But, $L_n = 1 + (V - p_V)S(n,1,\mathbf{p},1) - S(n,0,\mathbf{p},1-p_V) - (V-1)S(n,0,p,1)$ and

$$L_n = 1 + n\frac{V-p_V-(1-p_V)\ln(1-p_V)}{-\sum_{i=1}^{V} p_i \ln p_i} - \frac{V}{V-1} + f_{-1,0}(n) + O(1)$$

where $f_{-1,0}(n) = (V-1-p_V)f_0(n-1) - (V-1)f_{-1}(n)$. Then (4) implies

$$\lambda_{\max}(\mathbf{p},V)^{-1} = \limsup_{n\to\infty} \frac{L_n}{n} = \frac{-\sum_{i=1}^{V} p_i \ln p_i}{V-p_V-(1-p_V)\ln(1-p_V)} + r_2 \tag{57}$$

where $r_2 = \limsup_{n\to\infty} f_{-1,0}(n)$, and r_2 is much smaller then the leading factor in (57), however, for bigger V r_2 is not negligible from the numerical point of view. Nevertheless, the leading factor is responsible for qualitative properties of the algorithm. Optimizing it with respect to $\mathbf{p}$ we may prove that (57) is maximized iff $p_1 = p_2 = \cdots = p_{V-1} = p$ and p_V satisfy the following equation

$$(V-p_V)(\ln p - \ln p_V + V - 2) + \ln p_V \cdot \ln(1-p_V) - (V-2)(1-p_V)\ln(1-p_V) = 0 \tag{58}$$

For V=2 numerical solution of (58) yields the single root $p_2 = 0.5825$, and $\lambda_{\max}(2)$ is then 0.38126. Direct search over the exact formula for $\lambda_{\max}$ gives 0.5825 and 0.3808 , respectively. We have found also that (57) is optimized in a set of real numbers for $V^* = 2.08$ with $\lambda_{\max} = 3.8208$.

4.3 ISA algorithm with multibit overhead

For (6) we must substitute in (9) b=V, m=1, $p_1 = p = 1/V$, $p_2 = q = 1-p$. Moreover, $a_n = (1+\beta)[1-Vq^n]$, $\hat{a}_n = -(1+\beta)Vp^n$, $L_0 = L_1 = 1+\beta$ and by (18)

$$L_n = (1+\beta) + (1+\beta)p \sum_{k=2}^{n} (-1)^k \frac{k-p^{k-1}}{p-p^{k-1}}, \quad n \ge 0$$

and by (38)

$$L_n = (1+\beta)\, n\, \frac{1+\ln V}{\ln V} - \frac{1+\beta}{V-1} + (1+\beta)p\, f_0(n-1) - (1+\beta)f_{-1}(n) + O(1) \tag{59}$$

Before we present stability analysis let us note that in this case we are able to present explicit formula for $f_{r-1}(n)$ defined in (34), Indeed, for m=1 equation (32) with $d_1 = p$ possesses the following solution

$$z_k^r = r-1+2\pi ik/\ln p \ , \quad k=1,\pm 1,\pm 2,... \quad r \geq 0 \tag{60}$$

Then, the function $f_{r-1}(n)$ is

$$\begin{aligned} f_{r-1}(n) &= \frac{-1}{lnp} \sum_{\substack{k=-\infty \\ k\neq 0}}^{\infty} \Gamma(r-1+2\pi ik/lnp)\exp\left[-2\pi ik \ \log_p n\right] \\ &= \frac{-2}{lnp} \sum_{k=1}^{\infty} \mathbf{Re} \ \Gamma(r-1+2\pi ik/lnp)\exp\left[-2\pi ik \log_p n\right] \end{aligned} \tag{61}$$

where **Re**(z) is the real part of z. This function was studied by Knuth [11] (see also [10]). In particular, the following properties may be established.

(P1) $f_r(n)$ is a periodic function of $\log_p n$. Indeed, $f_r(np) = f_r(n)$.

(P2) $f_r(n)$ is bounded. This is proved by using the following properties of $\Gamma(z)$ [8], [19]

$$|\Gamma(it)|^2 = \pi/(t \ \sinh \pi t) \ , \quad \Gamma(z+1) = z\Gamma(z).$$

(P3) For any fixed a $f_r(n-a) = f_r(n) + O(n^{-1})$, since $\log(n-a) = \log n + \log(1-\frac{a}{n}) = \log n + O(n^{-1})$.

In particular, property (P1) implies that L_n/n as $n \to \infty$ does not converge to any point, but it has a tiny oscillation [10]. In fact, property (P2) tells us how tiny the oscilation is. For example, Knuth [11] has computed that for p=0.5 $f_{-1}(n) < 1.725\ 10^{-7}$, p=0.2 $f_{-1}(n) < 8.5\ 10^{-4}$, however, for p =1/16 $f_{-1} < 0.0032$ which is not quite negligible from the numerical point of view.

Maximum throughput λ_{max} for the algorithm is computed according to (8). Note that $F(\mu) = H(\mu) = h(\mu) + L_0$ and $h(\mu)$ is a solution of (39) with $\alpha = V$ and p =1/V ($\alpha p = 1$). But, $H(\mu)$ is given by (44) with $f_k = (1+\beta)(k - Vp^k)$, that is

$$H(\mu) = (1+\beta)\left[1 + \sum_{k=2}^{\infty} (-1)^k \frac{(k - Vp^k)\,z^k}{k!(1-p^{k-1})}\right] \tag{62}$$

In Table 1 for $\beta = V/1024$ we compare the optimal value μ_{opt} of μ and maximum throughput λ_{max} found by direct search over (8) with L_n computed according to the recurrence (6), with optimal value μ^* and λ^*_{max} evaluated according to (8) and series approximation (62). The table shows very good accuracy between μ_{opt}, μ^*, λ_{max} and λ^*_{max}, however, it indicates also that optimal value of μ lies between 2 and 8 for $4 \leq V \leq 16$. This implies that neither small value approximation nor asymptotic approximation may give a good approach of the maximum throughput. Nevertheless, by (55) $H(\mu)$ is $H(\mu)=1+\beta+S(\mu,0,1-1/V)+S(\mu,0,1)+O(1)$, and using properties (P1)- (P3) we find after some algebra

$$H(\mu)=(1+\beta)\mu\{(1+lnV)/lnV - V^{-1}[f_0(\mu)-f_{-1}(\mu)]\}+(1+\beta)(V-1)/lnV+2(1+\beta)+O(1)$$

Using this in (8) and finding supremum over μ we are able to compute λ_{max}. However, these computation are neither much more simpler than the ones used in (62) (numerical point of view) nor they give better insight into the algorithm behavior (qualitative point of view). But this might be relaxed if we ignore the fluctuating terms $f_0(\mu)$ and $f_{-1}(\mu)$ which turns out to be much smaller then the leading factors. This approach is acceptable at least for qualiatative analysis of the algorithm. Then, the supremum of $\mu/H(\mu)$ is reached for $\mu=\infty$ and we find the following approximation for λ_{max}

$$\lambda_{max}^{\infty} \stackrel{def}{=} \frac{lnV}{(1+\beta)(1+lnV)} \tag{63}$$

In Table 1 we compare λ_{max}^{∞} with λ_{max}. It suggests that approximation (63) is acceptable only for bigger values of V, however, the advantage of (63) lies in its simplicity.

Table 1

V	μ_{opt}	λ_{max}	μ^*	λ^*_{max}	λ_{max}^{∞}
4	2.3	0.6144	2.3	0.6144	0.578
8	5.0	0.6870	5.0	0.6870	0.670
16	7.2	0.7456	7.2	0.7456	0.723

In particular, (63) shows the impact of V and β on the maximum throughput. Moreover, other quantities of interest may be evaluated through $H(\mu)$ which represents unconditional average length of CRI (see [5]). Then, small value and asymptotic approximations can be used.

5. CONCLUSIONS

Three conflict resolution algorithms were considered. Two of them slightly generalized Capetanakis - Tsybakov - Mikhailov stack algorithm, while the third one is an interval-searching algorithm with multibit overhead. To analyze them we have introduced a recurrence equation which was solved (a closed form expression), and we have presented an asymptotic approximation for it. In addition, small value and asymptotic approximations for a solution of a functional equation associated with the recurrence were considered. These general studies were applied in Section 4 to evaluate maximum throughput for the three CRA algorithms.

The analysis of Section 3 is not only restricted to throughput evaluation of the above three conflict resolution algorithms. For example, a class of tree-type CRA algorithms considered in [10] might be analyzed in a uniform way using the studies from Section 3. Moreover, more sophisticated performance evaluation of some CRA algorithms may be done through analysis of the recurrence and functional analysis introduced in this paper (see [5], [9], [13]). In addition, many problems in algorithm design and analysis of computer science field may be reduced to a solution of recurrence (9),e.g. for radix exchange sorting [11], analysis of tries [7] and so on. More examples the reader may find in [7] and [11].

APPENDIX

We prove formulas (37b) and (37c). For (37b) we have to find the residue of

$$g_0(z) = \frac{\Gamma(z)(nc)^{z-1}}{D - \sum_{i=1}^{m} d_i^{-z}} \tag{A1}$$

for $z_0 = -1$. Note that $z_0 = -1$ is a pole of $\Gamma(z)$ as well as the zero of the denominator. To handle it we first determine an expansion of $\Gamma(z)$ around $z = -1$. Let $w = z+1$. Then [8], [19]

$$\begin{aligned}
&\Gamma(z+2) = \Gamma(z)z(z+1) \\
&\Gamma(w+1) = 1 - \gamma w + O(w^2) \\
&\frac{1}{w-1} = -1 - w + O(w^2),
\end{aligned}$$

and

$$\Gamma(z) = \frac{\Gamma(w+1)}{w(w-1)} = -w^{-1} + (\gamma-1) + O(w) \tag{A2}$$

Moreover,

$$(nc)^{-1-z} = 1 - w ln(nc) + O(w^2) \tag{A3}$$

$$\frac{1}{D - \sum_{i=1^m} d_1^{-z}} = -\frac{w^{-1}}{h_d} + f_0 + O(w) \tag{A4}$$

where $h_d = -\sum_{i=1}^{m} d_i \ln d_i$. To find f_0 we note that

$$f_0 = \lim_{z \to -1} \left\{ \left[D - \sum_{i=1}^{m} d_i^{-z} \right]^{-1} + \frac{h_d}{(z+1)} \right\},$$

and

$$D - \sum_{i=1}^{m} d_i^{-z} = -(z+1)h_d - \frac{(z+1)^2}{2} \sum_{i=1}^{m} d_i \, (ln \; d_i)^2 + O(w^3)$$

Let $h_d^{(2)} = \sum_{i=1}^{m} d_i (\ln d_i)^2$. Then

$$f_0 = \lim_{w \to 0} \frac{w^2(h_d^{(2)} + O(w^3))}{2[w^2 h_d^2 + O(w^3)]} = \frac{h_d^{(2)}}{2h_d^2}$$

The residue at $z_0 = -1$ is the coefficient of w^{-1} in the product of (A2), (A3) and (A4), and is given by (37b).

The proof of (37c) is similar, however, now the following expansions must be considered

$$\Gamma(z) = z^{-1} - \gamma + O(z)$$

$$(nc)^{-z} = 1 - z \; \ln nc + O(z^2)$$

$$\frac{1}{D - \sum_{i=1}^{m} d_i^{1-z}} = -z^{-1}/h_d + h_d^{(2)}/2h_d^2 + O(z)$$

The product of these gives the coefficient at z^{-1} which is the desired residue presented by (37c).

□

ACKNOWLEDGEMENTS

The author is grateful to an anonymous referee for his valuable comments.

REFERENCES

[1] BERGER, T.; The Poisson multiple-access conflict resolution problem. In *Multiuser Communications (Ed. G. Longo) CISM Course and Lectures* No. 265, pp. 1–27, *Springer-Verlag* 1981.

[2] BERGER, T., HUANG, J.; A conflict resolution algorithm with multibit feedback. *To be published.*

[3] CAPETANAKIS, J.; Tree algorithms for packet broadcast channels. *IEEE Trans. Inf. Th., IT-25*, (1979) pp. 505–515.

[4] CAPETANAKIS, J.; Generalized TDMA: The multiaccessing tree protocol. *IEEE Trans. Comm.*, COM-27, (1979) pp. 1476–1587.

[5] FAYOLLE, G., FLAJOLET, Ph., HOFRI, M., JACQUET P.; Analysis of stack algorithm for random multiple-access communication. *IEEE Trans. Inf. Theory* , vol. IT-31, 2, (1985), pp.244-254.

[6] GALLAGER, R.; Conflict resolution in random access broadcast networks. *Proceedings AFOSR Workshop in Communication Theory and Applications*, (1978) pp. 74–76.

[7] GONNET G.; Handbook of algorithms and data structures. *Addison-Wesley*, (1984).

[8] HENRICI, P.; Applied and computational complex analysis. vol. I, II. *John Wiley & Sons*, New York 1977.

[9] HOFRI, M.; Stack algorithms for collision-detecting channels and their analysis: A limited survey. *Proceedings Intern. Seminar on Modelling and Performance Evaluation Methodology*, Paris 1983, vol. I, pp. 53–78.

[10] KAPLAN M., GULKO, E.; Analytic properties of multiple-access trees. *INRS-Telecommunications* TR 83-17, 1983.

[11] KNUTH, D.; The art of computer programming, vol. III, Sorting and Searching. *Addison-Wesley* (1973).

[12] MASSEY, P.; Collision resolution algorithms and random access communications. In *Multiuser Communications (Ed. G. Longo) CISM Courses and Lectures* No. 265, *Springer-Verlag* (1981) pp. 73–137.

[13] MATHYS, P., FLAJOLET P.; Q-ary collision resolution algorithms in random-access systems with free and blocked channel access. *IEEE Trans. Inf. Theory*, vol. IT-31, 2, (1985), pp.217-243.

[14] PAKES , A.; Some conditions for ergodicity and recurrence of Markov chains. *Operat. Res.*, 17, (1969), pp.1058-1061.

[15] RIODAN J.; Combinatorial identities. *John Wiley & Sons*, New York (1968).

[16] TSYBAKOV B., MIKHAILOV, V.; Free synchronous packet access in a broadcast channel with feedback. *Prob. Inf. Trans.*, 14, 4, (1978) pp. 259–280.

[17] TSYBAKOV B., MIKHAILOV, V.; Random multiple packet access: Part-and-Try algorithm. *Prob. Inf. Trans.*, 16, 4, (1980) pp. 305–317.

[18] TWEEDIE, R.; Criteria for classifying general Markov chains. *Adv. Appl. Prob.*, 8, (1976), pp.737-771.

[19] WHITTAKER, E., WATSON, G.; A course of modern analysis. *Cambridge* (1935).