Discrete Applied Mathematics, **39** (1992), 207–229

BIRTHDAY PARADOX, COUPON COLLECTORS, CACHING ALGORITHMS AND SELF-ORGANIZING SEARCH

Philippe Flajolet
INRIA, Rocquencourt
78150 Le Chesnay (France)

Danièle Gardy
LRI, Université Paris-Sud
91405 Orsay (France)

Loÿs Thimonier
Université de Picardie
33 rue Saint Leu
80039 Amiens (France)
and
LRI, Université Paris-Sud

Abstract: *This paper introduces a unified framework for the analysis of a class of random allocation processes that include: (i) the birthday paradox; (ii) the coupon collector problem; (iii) least-recently-used (LRU) caching in memory management systems under the independent reference model; (iv) the move-to-front heuristic of self-organizing search. All analyses are relative to general non uniform probability distributions.*

Our approach to these problems comprises two stages. First, the probabilistic phenomena of interest are described by means of regular languages extended by addition of the shuffle product. Next, systematic translation mechanisms are used to derive integral representations for expectations and probability distributions.

1. Some Random Allocation Problems

We present in this paper a unified treatment for a number of related probabilistic allocation problems. The problems that we consider will be defined in detail in later sections, but we offer here an informal description.

1. *Birthday Paradox* [BP]: One needs on the average 24 people to discover two that have the same birthday, assuming all birth dates to be equally distributed over the days in the year. Generalizations of this problem concern non-uniform distributions, and multiple "hits".

2. *Coupon Collector Problem* [CCP]: A company issues coupons of different types, each type having a certain probability of being issued. The coupon collector problem asks for the expected number of coupons that need to be gathered before a full collection is obtained.

3. *Least-Recently-Used Caching* [LRU]: Caching algorithms aim at maintaining fast access to a large number of items by keeping a small "cache" that may be addressed quickly. The classical problem of cache analysis consists in determining the steady state probability of a cache "fault" when items are accessed with a fixed, not necessarily uniform, distribution. The LRU caching strategy consists in applying replacement, when needed, to the oldest element in the cache (the "least recently used" element).

4. *Self-Organizing Search, 'Move-To-Front' Rule* [MTF]: If a list of items is to be searched sequentially, the optimum arrangement is by decreasing order of access probabilities. Self-organizing strategies aim at minimizing the access time to items, when the underlying probability distribution of item accesses is unknown. A good heuristic is the so-called "Move-To-Front" heuristic, under which an element is moved to the front of the list when it is accessed. The problem is to determine the steady state cost of this method.

It can be recognized that these four problems have a common flavor: A sequence of elements from a finite universe is drawn at random, according to some probability distribution; when an element arrives, a certain action is taken depending on the different elements present in the system, and a corresponding cost function has to be analyzed. As we shall see, the four problems go by pairs: CCP is a specialization of BP, and MTF resembles a particular case of LRU.

Our methodological approach to these analyses is related to symbolic methods in combinatorial analysis. It can be described as follows: (i) Determine a proper specification of the underlying combinatorial process in terms of formal languages. (ii) Use systematic translation mechanisms to derive generating function expressions for quantities of interest. The class of formal languages relevant to our analyses is the class of regular languages, and the "shuffle" product plays a particularly important rôle in the formal descriptions that we encounter. In this way, we obtain expressions that are combinations of rational operations (corresponding to usual regular language operations) and Laplace transform integrals (arising from shuffle products).

We provide exact solutions to these four problems under a general probability distribution for items (birth dates, coupons, memory references, keys), only assuming independence. Because of the occurrence of shuffle products, the solutions are naturally expressed as integrals, from which symmetric function expressions can be derived.

It can be seen that the problems considered here are of a *Markovian* nature. Though we make some occasional use of Markovian properties, this observation is of little help for explicit computations since the associated Markov chains tend to have an exponential number of states. Furthermore, our analyses lead to integral representations for quantities of interest that bear no resemblance to the expressions usually obtained by standard Markovian analysis [Feller 1968]. These integral representations are computationally useful (see Section 7). For instance, a Markovian analysis of a typical cache problem would require roughly time 10^{180} and space 10^{60}, and the time complexity would only decrease to about 10^{40} using symmetric function expressions that resemble a summation over all possible cases. Instead our integral forms (Theorem 5) can be estimated using about $10^7/10^8$ elementary function evaluations, which is achievable on a medium size computer.

Section 2 introduces the necessary background regarding languages and probabilities. The succeeding sections (Sections 3-6) present the analyses of the four problems that we have described. Section 7 concludes with a brief discussion of potential applications of our integral representations.

A preliminary version of this paper has been presented at the 15th ICALP Colloquium [Flajolet *et al.* 1988].

2. Formal Languages and Probabilities

In this section, we recall general tools for translating formal specifications by regular languages into counting and probability estimates. General references on this subject are [Salomaa, Soittola 1978], [Goulden, Jackson 1983], [Comtet 1974], [Stanley 1986] and [Wetherell 1980].

Regular Languages. Let $\mathcal{A} = \{a_1, a_2, \ldots, a_m\}$ be a fixed set called the *alphabet* whose elements are the *letters*; $\mathcal{A}^*$ represents the set of all finite sequences –called *words* or *strings*– of $\mathcal{A}$. A *language* is any subset of $\mathcal{A}^*$. Let L, L_1, L_2 be languages. The union of L_1 and L_2 is denoted by $L_1 + L_2$. The (catenation) product of L_1 and L_2, denoted by $L_1 \cdot L_2$, is defined as

$$L_1 \cdot L_2 = \{w_1 w_2 \ / \ w_1 \in L_1, \ w_2 \in L_2\}$$

and the "star" operation L^* is obtained by forming all possible sequences from elements of L,

$$L^* = \{\epsilon\} + L + (L \cdot L) + (L \cdot L \cdot L) + \cdots$$

with ϵ denoting the empty word.

The class of *regular languages* is classically defined as the smallest class of languages containing the finite sets and closed under the three operations of union, product and star. Regular languages are also closed under a fourth operation, crucial to our analysis, the *shuffle* product [Lothaire 1983]. If w_1 and w_2 are words, their shuffle, denoted† by $(w_1 \sqcup\!\sqcup\, w_2)$ consists of the set of all words obtained by mixing in all possible ways letters of w_1 and w_2 while preserving their order inside w_1 and w_2. It is defined recursively by

$$(av_1 \sqcup\!\sqcup\, bv_2) = a(v_1 \sqcup\!\sqcup\, bv_2) \cup b(av_1 \sqcup\!\sqcup\, v_2),$$

with $(v \sqcup\!\sqcup\, \epsilon) = (\epsilon \sqcup\!\sqcup\, v) = \{v\}$. The shuffle of two languages L_1 and L_2 is

$$L_1 \sqcup\!\sqcup\, L_2 = \bigcup_{\substack{w_1 \in L_1 \\ w_2 \in L_2}} (w_1 \sqcup\!\sqcup\, w_2).$$

For instance, $(ab \sqcup\!\sqcup\, cd) = \{abcd, acbd, acdb, cabd, cadb, cdab\}$.

Generating Functions. The important property, as far as counting and estimating probabilities is concerned, is that these operations have direct translations into generating functions. If L is a language, we let $\ell_{n_1,\ldots,n_m}$ be the number of words in L that have n_1 occurrences of letter a_1, ..., n_m occurrences of a_m. The *multivariate generating function* of L is

$$\ell(z_1, \ldots, z_m) = \sum_{n_1,\ldots,n_m \geq 0} \ell_{n_1,n_2,\ldots,n_m} z_1^{n_1} z_2^{n_2} \cdots z_m^{n_m}.$$

† This symbol is a Russian letter *ša*, an abbreviation for *shuffle* often used in combinatorics on words.

NOTATION: Given a generating function

$$f(x,y,z) = \sum_{m,n,p} f_{m,n,p} x^m y^n z^p$$

we use $[x^m y^n z^p] f(x,y,z)$ to denote the coefficient of $x^m y^n z^p$ in f:

$$[x^m y^n z^p] f(x,y,z) = f_{m,n,p}. \qquad \blacksquare$$

We now assume that a fixed "weight" distribution (which we shall specialize in a moment to be a probability distribution) $\mathbf{p} = (p_1, p_2, \ldots, p_m)$ over $\mathcal{A}$ is given, so that p_i is the weight of letter a_i. The weight is extended multiplicatively to words, the weight of $w = a_{j_1} a_{j_2} \cdots a_{j_n} \in \mathcal{A}^n$ being taken as

$$\pi[w] = p_{j_1} p_{j_2} \cdots p_{j_n}.$$

The function

$$\ell(p_1 z, p_2 z, \ldots, p_m z) = \sum_{n_1,\ldots,n_m} \ell_{n_1,\ldots,n_m} p_1^{n_1} \cdots p_m^{n_m} z^{n_1+\cdots+n_m} = \sum_{w \in \mathcal{A}^*} \pi[w] z^{|w|} \tag{1}$$

is called the *ordinary generating function* (OGF) of language L (with respect to weight $\mathbf{p}$) and is denoted by $\ell(z)$. The *exponential generating function* (EGF) is similarly defined, with $z^n/n!$ replacing z^n:

$$\sum_{n_1,\ldots,n_m} \ell_{n_1,\ldots,n_m} p_1^{n_1} \cdots p_m^{n_m} \frac{z^{n_1+\cdots+n_m}}{(n_1+\cdots n_m)!} = \sum_{w \in \mathcal{A}^*} \pi[w] \frac{z^{|w|}}{|w|!} \tag{2}$$

and we denote it by $\hat{\ell}(z)$.

When the weight function satisfies $\mathbf{p} = (1, 1, \ldots, 1)$, we have the classical ordinary and exponential generating functions from combinatorial analysis (see e.g. [Comtet 1974], [Goulden, Jackson 1983], [Stanley 1986]), and $[z^n]\ell(z)$ is the number of words of length n in L. From now on, we specialize the weight distribution to be a probability distribution over $\mathcal{A}$, so that $\sum_{i=1}^m p_i = 1$. This defines in the usual way the product probability measures on $\mathcal{A}^n$ and $\mathcal{A}^\infty$ (the set of infinite sequences): $\pi[w]$ is the probability of word w in $\mathcal{A}^n$ as well as the probability of $w \cdot \mathcal{A}^\infty$ in $\mathcal{A}^\infty$. Accordingly, $[z^n]\ell(z)$ represents the probability that a random word in $\mathcal{A}^n$ be in L. It is also the probability that a random word of $\mathcal{A}^\infty$ belongs to $(L \cap \mathcal{A}^n) \cdot \mathcal{A}^\infty$ when L is "prefix-free" (i.e. no prefix of a word of L is in L).

As is well known, ordinary and exponential generating functions are related by the Laplace-Borel transform,

$$\ell(z) = \int_0^\infty \hat{\ell}(zt) e^{-t}\, dt, \tag{3}$$

as follows from the classical relation

$$\int_0^\infty t^n e^{-t}\, dt = n!.$$

In the sequel, we adhere to the notational convention of representing a language L, and its generating functions $\ell(z)$, $\hat{\ell}(z)$ by the same letters. With this convention, we can state:

THEOREM 0. *When they operate unambiguously on their arguments, the operations of union, product, star and shuffle product translate into generating functions:*

$$\begin{array}{llll}
L = L_1 + L_2 & \Longrightarrow & \ell(z) = \ell_1(z) + \ell_2(z) & (a) \\
L = L_1 \cdot L_2 & \Longrightarrow & \ell(z) = \ell_1(z) \cdot \ell_2(z) & (b) \\
L = L_1^* & \Longrightarrow & \ell(z) = (1 - \ell_1(z))^{-1} & (c) \\
L = L_1 ш L_2 & \Longrightarrow & \hat{\ell}(z) = \hat{\ell}_1(z) \cdot \hat{\ell}_2(z). & (d)
\end{array}$$

In essence, an operation on languages is unambiguous if every word of the resulting language is obtained only once. Thus, an unambiguous union is one that operates on disjoint languages; product $L = L_1 \cdot L_2$ is unambiguous iff each $w \in L$ has a unique decomposition as $w_1 w_2$; a star operation is unambiguous if the defining unions and products are. Finally, a shuffle product $L = L_1 ш L_2$ is unambiguous when languages L_1 and L_2 are subsets of A_1^* and A_2^* where $A_1, A_2 \subset \mathcal{A}$ satisfy $A_1 \cap A_2 = \emptyset$.

The first three cases (a–c) are the basis of the classical Chomsky-Schützenberger (1963) theorem: *Every regular language has a rational generating function.* The result for the shuffle product is a classical one in the context of word enumerations [Lothaire 1983]; its proof is based on the observation that

$$\ell_n = \sum_{n_1+n_2=n} \binom{n}{n_1, n_2} \ell_{1,n_1} \ell_{2,n_2} \qquad \text{or} \qquad \frac{\ell_n}{n!} = \sum_{n_1+n_2=n} \frac{\ell_{1,n_1}}{n_1!} \frac{\ell_{2,n_2}}{n_2!},$$

where the multinomial coefficient counts the number of possible shuffles on words of lengths n_1 and n_2 and $\ell_n = [z^n] l(z)$ etc.

Finally, in many cases, we not only need to determine probabilities of events but also analyze distributions or expected values of auxiliary parameters. This is achieved in the usual way by introducing a further variable in generating functions. Let $\phi: \mathcal{A}^* \to \mathbf{N}$ be an integer valued parameter of words. Then the OGF and EGF of L *with variable v "marking" parameter ϕ* are defined as

$$l(z; v) = \sum_{w \in \mathcal{A}^*} \pi[w] z^{|w|} v^{\phi(w)}, \qquad \text{and} \qquad \hat{l}(z; v) = \sum_{w \in \mathcal{A}^*} \pi[w] \frac{z^{|w|}}{|w|!} v^{\phi(w)}.$$

There is a direct extension of Theorem 0 to these bivariate generating functions. Let L and M be two languages related by $L = M^*$, where the star operation is unambiguous; a parameter $\phi: M \to \mathbf{N}$ can be extended additively to L by

$$\phi(w_1 w_2 \cdots w_r) = \phi(w_1) + \phi(w_2) + \cdots + \phi(w_r) \qquad \text{where} \qquad w_j \in M.$$

If $l(z; v)$ and $m(z; v)$ are the generating functions of L and M with v marking ϕ, then

$$\begin{aligned}
l(z; v) &= \sum_{w \in L} \pi[w] z^{|w|} v^{\phi(w)} \\
&= \sum_{r \geq 0} \sum_{w_1, \cdots w_r \in M} \pi[w_1] \cdots \pi[w_r] z^{|w_1| + \cdots |w_r|} v^{\phi(w_1) + \cdots + \phi(w_r)} \\
&= \frac{1}{1 - m(z; v)}.
\end{aligned}$$

Probabilities. From Theorem 0, there is a general procedure to determine the generating function of a language defined by a combination of the four basic operations. This makes it possible to analyze "mechanically" probability distributions of combinatorial parameters described by regular languages. (See for instance [Goulden Jackson 1983], [Greene 1983], [Flajolet 1988] for other approaches). The expressions obtained are a combination of rational operations and Laplace transform integrals* wherever shuffle products appear.

3. Birthday Paradox

The alphabet $\mathcal{A}$ represents here the dates in a year with m days, and p_i is the probability of date $a_i \in \mathcal{A}$. We consider the following generalization of the birthday problem:

BP. Determine the expectation of the number B_j of elements that need to be drawn from $\mathcal{A}$ (with replacement) till we first encounter j distinct elements that are each repeated at least k times (i.e. the waiting time till the j-th different-letter occurrence of a k-hit).

The case $k = 2$, $j = 1$ is the classical birthday problem. Klamkin and Newman [1967] have given an integral formula for $j = 1$ (first hit) and general k in the *uniform case* where $p_i = 1/m$.

Theorem 1. *The expectation* $\mathrm{E}\{B_j\}$ *of the time for obtaining* j *different-letter occurrences of a* k*-hit under a general probability distribution* $\{p_i\}_{i=1}^m$ *is given by*

$$\mathrm{E}\{B_j\} = \sum_{q=0}^{j-1} \int_0^\infty [u^q] \left(\prod_{i=1}^{m} \left(e_{k-1}(p_i t) + u(e^{p_i t} - e_{k-1}(p_i t)) \right) \right) e^{-t}\, dt \tag{4}$$

where $e_k(t)$ *represents the truncated exponential*

$$e_k(t) = 1 + \frac{t}{1!} + \frac{t^2}{2!} + \cdots + \frac{t^k}{k!}.$$

Before going into the proof we mention an immediate corollary:

Corollary 1. *The expected time of a first* k*-hit* $\mathrm{E}\{B_1\}$ *is given by*

$$\mathrm{E}\{B_1\} = \int_0^\infty \left(\prod_{i=1}^{m} e_{k-1}(p_i t) \right) e^{-t} dt \tag{5a}$$

In the equiprobable case ($p_i = 1/m$)*, we have*

$$\mathrm{E}\{B_1\} = \int_0^\infty \left(e_{k-1}(\frac{t}{m})\right)^m e^{-t} dt, \tag{5b}$$

and more generally

$$\mathrm{E}\{B_j\} = \sum_{q=0}^{j-1} \binom{m}{q} \left[\int_0^\infty \left(e_{k-1}(\frac{t}{m}) \right)^{m-q} \left(e^{t/m} - e_{k-1}(\frac{t}{m}) \right)^q e^{-t} dt \right]. \tag{5c}$$

---

* In the most general situation, direct and inverse transforms may occur when we go back and forth between ordinary and exponential generating functions. As pointed by a referee, for the applications discussed in the current paper, all shuffle products appear before all products, so that only direct Laplace transform integrals of type (3) appear in our formulæ. See however the two stack problem in [Flajolet 1986] that involves taking an inverse Laplace transform.

Equation (5b) is Klamkin and Newman's original result and though they do not state it explicitly, their approach could readily provide the more general result (5a). Equations (5c) and (4) are natural generalizations of Klamkin and Newman's integral formula.

PROOF. The proof decomposes into two stages: a preparatory probabilistic argument and a suitable regular language description of the problem.

We start by the (classical and easy) probabilistic argument. The random variable (RV) B_j is a first time of occurrence of a certain event in an infinite sequence of trials. It is thus a RV defined on $\mathcal{A}^\infty$ with the product measure. Let Y_n be the RV defined on $\mathcal{A}^n$ representing the number of k-hits (on different letters) in a sequence of n trials. Though the probability spaces are not the same, the two probability distributions are related by

$$\Pr\{Y_n \geq j\} = \Pr\{B_j \leq n\}. \tag{6}$$

To see it, introduce the language G_j consisting of words with at least j hits of multiplicity k. The first quantity in (6) is the probability that a random word of length n belongs to G_j, while the second one is the measure of $(G_j \cap \mathcal{A}^n) \cdot \mathcal{A}^\infty$. From this, the expectation of B_j is easily found:

$$\begin{aligned} \mathrm{E}\{B_j\} &= \sum_{n \geq 0} \Pr\{B_j > n\} \\ &= \sum_{n \geq 0} \Pr\{Y_n < j\} \\ &= \sum_{q=0}^{j-1} \left(\sum_{n \geq 0} \Pr\{Y_n = q\} \right) \end{aligned} \tag{7}$$

The first equation in (7) is a classical form for expectations of discrete random variables, and the second follows from the equivalence principle (6). The third equation only expresses the decomposition of $\Pr\{Y_n < j\}$ according to possible values of Y_n.

The problem is now reduced to estimating the inner sum in Eq (7). Let H_q be the language consisting of words with exactly q letters that occur at least k times (the other r letters occurring at most $k-1$ times), so that $H_q = G_q \setminus G_{q-1}$. With $\alpha^{<k} = \epsilon + \alpha + \alpha^2 + \cdots + \alpha^{k-1}$ and $\alpha^{\geq k} = \alpha^k \cdot \alpha^*$, language H_q is specified by

$$H_q = \bigcup_{I,J} \left(a_{i_1}^{\geq k} ш\, a_{i_2}^{\geq k} ш \cdots ш\, a_{i_q}^{\geq k} \right) ш \left(a_{j_1}^{<k} ш\, a_{j_2}^{<k} ш \cdots ш\, a_{j_r}^{<k} \right), \tag{8}$$

where the summation is over all sets I, J of cardinality q and $r = m - q$ such that

$$I = \{i_1, \ldots, i_q\},\ J = \{j_1, \ldots, j_r\} \qquad \text{with} \qquad I \cap J = \emptyset,\ I \cup J = \{1, 2, \ldots, m\}. \tag{9}$$

If α is a letter with probability σ, the EGFs of $\alpha^{<k}$ and $\alpha^{\geq k}$ are

$$e_{k-1}(\sigma z) \qquad \text{and} \qquad e^{\sigma z} - e_{k-1}(\sigma z).$$

Thus by Theorem 0, the EGF of H_q is (with I, J in the sum satisfying (9))

$$\hat{h}_q(z) = \sum_{I,J} \left((e^{p_{i_1} z} - e_{k-1}(p_{i_1} z)) \cdots (e^{p_{i_r} z} - e_{k-1}(p_{i_q} z)) \right), \left(e_{k-1}(p_{j_1} z) \cdots e_{k-1}(p_{j_r} z) \right)$$

and noting the general expansion

$$[u^q]\prod_{i=1}^{m}(\lambda_i u+\mu_i)=\sum_{I,J}(\lambda_{i_1}\cdots\lambda_{i_q})\cdot(\mu_{j_1}\cdots\mu_{j_r}),$$

we can express $\hat{h}_q(z)$ as

$$\hat{h}_q(z)=[u^q]\Phi(z,u)\qquad\text{where}\qquad \Phi(z,u)=\prod_{i=1}^{m}\left(e_{k-1}(p_it)+u(e^{p_it}-e_{k-1}(p_it))\right). \tag{10}$$

Now the OGF of H_q is given by the Laplace-Borel transform (3),

$$h_q(z)=\int_0^\infty [u^q]\Phi(zt,u)\,e^{-t}dt,$$

so that we have

$$\sum_{n\geq 0}\Pr\{Y_n=q\}=h_q(1)=\int_0^\infty [u^q]\,\Phi(t,u)\,e^{-t}dt, \tag{11}$$

and a combination of (11) and (7) yields the statement of Theorem 1. ■

The estimates of Theorem 1 are clearly symmetric functions of the $\{p_j\}$ and can sometimes be expressed in a reasonably pleasant form. Expanding the products in (5b) and evaluating the integrals, we find:

Corollary 2. *For the classical birthday paradox ($k=2$ and $j=1$), the expectation of waiting time is*

$$\mathrm{E}\{B_1\}=1+1!\,S_1+2!\,S_2+\cdots m!\,S_m,$$

where the S_r are elementary symmetric functions of the p_j

$$S_r=\sum_{j_1<j_2<\cdots<j_r}p_{j_1}p_{j_2}\cdots p_{j_r}. \tag{12}$$

For instance, with $m=3$ and $(p_1,p_2,p_3)=(a,b,c)$, we have

$$\mathrm{E}\{B_1\}=1+1!\,(a+b+c)+2!\,(ab+bc+ca)+3!\,abc,$$

and for general m and a uniform distribution ($p_i=1/m$),

$$\mathrm{E}\{B_1\}-1=1+\frac{m-1}{m}+\frac{(m-1)(m-2)}{m^2}+\cdots+\frac{(m-1)(m-2)\cdots 1}{m^{m-1}},$$

a sum that was studied by Ramanujan and shows in several analyses of algorithms. Hashing with linear probing [Knuth 1973, p.529] was Knuth's first analysis of an algorithm, on an afternoon of 1962. Following Ramanujan's treatment, Knuth [1968, p.112] uses it as an introduction to asymptotics by the Laplace method; it next appears [Knuth 1969, p.454] in the analysis of random mappings (related to random number generators). It is from this analysis that Pollard conceived his integer factorization algorithm (the "rho method", see page 608 of the second edition in 1981 of [Knuth 1969], and [Brent 1981]). The Ramanujan function finally arises in optimal caching

[Knuth 1985], the study of memory conflicts [Knuth, Rao 1975] and Union-Find algorithms [Knuth, Schönhage 1978].

On another register, Mase [1990] has carried out an exact analysis of the birthday problem with unequal occurrence probabilities from which he deduces an approximation model. His paper is also of interest since it is accompanied by numerical fitting on statistical data (based on Japanese surnames).

It may be of interest to conclude this section by noting that the Markov chain which corresponds to Corollary 2 has 2^m states.

4. Coupon Collector Problem

The alphabet $\mathcal{A}$ now represents the set of coupons, with p_i being the probability that coupon i is issued. The general coupon collector problem is the following:

CCP. Determine the expectation of the number C_j of elements that need to be drawn from $\mathcal{A}$ (with replacement), till one first obtains a collection with j different coupons.

Quantity $\mathrm{E}\{C_m\}$ is of particular interest since it represents the expected time to obtain a full collection. The solution of this problem in the equiprobable case is a classical exercise: One needs to draw 1 element to gather a collection of cardinality 1; then $m/(m-1)$ draws are necessary on the average to gather a new element etc. In this way, one finds

$$\mathrm{E}\{C_m\} = m\left(\frac{1}{m} + \frac{1}{m-1} + \frac{1}{m-2} + \cdots + \frac{1}{1}\right) = mH_m$$

where H_m is the m-th harmonic number. In the same vein, $\mathrm{E}\{C_j\} = m(H_m - H_{m-j})$. In the general case of a non-uniform probability distribution, we have:

THEOREM 2. *The expectation* $\mathrm{E}\{C_j\}$ *of the time necessary to gather a collection of* j *different items under a general probability distribution is given by*

$$\mathrm{E}\{C_j\} = \sum_{q=0}^{j-1} \int_0^\infty [u^q] \left(\prod_{i=1}^m \left(1 + u(e^{p_i t} - 1)\right)\right) e^{-t} dt, \tag{13a}$$

and for a full collection,

$$\mathrm{E}\{C_m\} = \int_0^\infty \left(1 - \prod_{i=1}^m \left(1 - e^{-p_i t}\right)\right) dt. \tag{13b}$$

PROOF. Form (13a) is just a specialization of formula (4) to the case $k=1$, and requires no further proof. To obtain (13b) from (13a) when $j = m$, introduce the function $\Phi(t,u) = \prod_{i=1}^m (1+u(e^{p_i t}-1))$ which is the special case for $k = 1$ of the Φ function of Eq (10). If we expand it as $\Phi(t,u) = \sum_{q=0}^m \varphi_q(t) u^q$, we get

$$\begin{aligned}\varphi_0(t) + \varphi_1(t) + \cdots + \varphi_{m-1}(t) =& \Phi(t,1) - \varphi_m(t) \\ =& e^t - \prod_{i=1}^m (e^{p_i t} - 1).\end{aligned} \qquad \blacksquare$$

Again symmetric function expressions are available in this case.

Corollary 3. *The expected time for a partial collection satisfies*

$$\mathrm{E}\{C_j\} = \sum_{q=0}^{j-1} (-1)^{j-1-q} \binom{m-q-1}{m-j} \sum_{|J|=q} \frac{1}{1-P_J} \qquad \textit{with} \qquad P_J = \sum_{j\in J} p_j, \tag{14a}$$

and for a full collection

$$\mathrm{E}\{C_m\} = \sum_{q=0}^{m-1} (-1)^{m-1-q} \sum_{|J|=q} \frac{1}{1-P_J}. \tag{14b}$$

For instance, when $m = 3$ and $(p_1, p_2, p_3) = (a, b, c)$, we find

$$E\{C_m\} = 1 - \frac{1}{1-a} - \frac{1}{1-b} - \frac{1}{1-c} + \frac{1}{1-a-b} + \frac{1}{1-b-c} + \frac{1}{1-c-a}.$$

Mean and variance estimates expressed as symmetric functions were obtained by Nath [1973].

For general m and a uniform distribution, the symmetric function expression reduces to

$$\frac{1}{m}\mathrm{E}\{C_m\} = \sum_{q=1}^{m} (-1)^{q-1} \binom{m}{q} \frac{1}{q},$$

a quantity otherwise well–known to be equal to H_m.

5. Least-Recently-Used Caching Algorithms

Caching algorithms are general purpose methods used to speed up access to a large collection of items stored on a slow device, by maintaining a small "cache" on a high speed device. An infinite sequence $w = w_1 w_2 w_3 \ldots$ of elements of $\mathcal{A}$, also called references, represents the items to be accessed at times $1, 2, 3, \ldots$. At any given time t, the cache contains a subset of size k of $\mathcal{A}$ where k is a fixed design parameter. Let K_t be the state of the cache at time t, assuming we are given an initial state K_0 of the cache at time 0. A *cache algorithm*, or page replacement algorithm, specifies the transition from K_{t-1} to K_t when reference w_t arrives:

$$K_t = \begin{cases} K_{t-1} & \text{if} \quad w_t \in K_{t-1} & (15a) \\ K_{t-1} - \{x_t\} + \{w_t\} & \text{if} \quad w_t \notin K_{t-1} & (15b) \end{cases}$$

In case (b), the selected item $x_t \in K_t$ is specified by the cache policy and we say that a *fault* occurred: Since the element to be accessed w_t is not in the cache, an element x_t is removed from the cache and w_t is inserted. In the other case (a), the element is found in the cache and we say that a *hit* occurred.

The *independent reference model* is the probabilistic model under which the references are independent random variables with a common distribution $\{p_i\}_{i=1}^{m}$ unknown to the algorithm. The cache analysis problem consists in determining the steady state probability of a page fault under this model. We consider here a well-known and important cache algorithm, the least-recently-used (LRU) algorithm:

LRU. In the replacement rule (15), select as $x_t \in K_{t-1}$ the element that is the oldest to have been last referenced.

Other important caching algorithms have been known for a long time, and the reader can refer to Smith's paper [1982] for an extensive survey of practical issues involved in cache design.

RAND. In the random replacement algorithm, x_t is chosen randomly to be any of the k elements present in the cache with probability $1/k$. (The algorithm is non-deterministic).

FIFO. The first-in-first-out algorithm choses as x_t the element that is the oldest to have entered the cache. This differs from LRU, where an element that is frequently accessed is very likely to remain in the cache for a long period of time.

OPT. The "optimal algorithm" has the peculiarity of depending on the future: The element x_t to be replaced is the one whose next access is the most remote in the future. The optimality of this strategy has been discovered by Belady [1966]. Surprising as it may seem, OPT can turn out to be a practical algorithm in certain contexts where a machine is driving another device's cache, as shown by Fuchs and Knuth [1985].

Several analytic results are known under the independent reference model, and a good discussion is given in Chapter 6 of [Coffman, Denning 1973]. Algorithm RAND has long been recognized to behave poorly since it does not discriminate between frequently accessed items and others. FIFO has been analyzed by King [1971], and Gelenbe [1973] later showed that the page fault probability of RAND is the same as that of FIFO. LRU is known to perform fairly well in many practical situations since it tends to keep frequently accessed elements in the cache. King also presents symmetric function expressions for its page fault probabilities (see our Corollary 4 below). Both FIFO and LRU are analyzed in [Aven, Boguslavsky, Kogan 1976], while Fagin and Price [1978] discuss the complexity of evaluating the analytic formulae in both cases and propose an interesting simulation scheme of low complexity for LRU. (Fagin [1977] also derived asymptotic approximations to LRU miss ratio.) OPT provides an upper bound on the efficiency of any general purpose caching algorithm. Little is known yet about its performances under non-uniform models: Knuth [1985] gave a partial analysis under a uniform probability distribution ($p_i = 1/m$) and showed that the page fault probability is $1 - O(\sqrt{k/m})$.

There is an alternative description of LRU caching by a *move-to-front* rule. Assume at each time the cache is kept as a *sequence* of elements arranged in order of latest reference time, so that $K_t = (c_t^{(1)}, c_t^{(2)}, \ldots, c_t^{(k)})$. Element $c_t^{(1)}$ is the last referenced, so that $c_t^{(1)} = w_t$ etc. Then the replacement rule (15) at time t is simply:

- In case of a page fault, eliminate element $c_{t-1}^{(k)}$ from the cache, shift all other elements down one position and prepend w_t,

$$K_t = \big(w_t, c_{t-1}^{(1)}, c_{t-1}^{(2)}, \ldots, c_{t-1}^{(k-1)}\big). \tag{16a}$$

- If there is no page fault and the referenced element w_t is in j-th position in the cache, shift down by one position the first $j-1$ elements and put w_t in the first position,

$$K_t = \big(w_t, c_{t-1}^{(1)}, \ldots, c_{t-1}^{(j-1)}, c_{t-1}^{(j+1)}, \ldots, c_{t-1}^{(k)}\big). \tag{16b}$$

For instance, if $\mathcal{A} = \{a, b, c, d, e\}$ and $w = badacedead\ldots$, with $k = 3$ and the cache initialized as $K_0 = (a, b, c)$, we have the sequence of transitions

$$abc \underset{b}{\overset{\circ}{\Rightarrow}} bac \underset{a}{\overset{\circ}{\Rightarrow}} abc \underset{d}{\overset{*}{\Rightarrow}} dab \underset{a}{\overset{\circ}{\Rightarrow}} adb \underset{c}{\overset{*}{\Rightarrow}} cad \underset{e}{\overset{*}{\Rightarrow}} eca \underset{d}{\overset{*}{\Rightarrow}} dec \underset{e}{\overset{\circ}{\Rightarrow}} edc \underset{a}{\overset{*}{\Rightarrow}} aed \underset{d}{\overset{\circ}{\Rightarrow}} dae$$

where $\underset{x}{\overset{*}{\Rightarrow}}$ and $\underset{x}{\overset{\circ}{\Rightarrow}}$ represent transitions with and without page faults, x being the element referenced.

This presentation of LRU caching has also the merit of showing that LRU caching under the independent reference model is a *Markov chain* with $k!\binom{m}{k}$ states, each state being an ordered combination of k elements amongst m. It is also clear that this Markov chain is irreducible and aperiodic. Thus, by standard Markov chain theory, the long run (stationary) probability of a cache fault is well defined and is independent of the initial state of the cache.

THEOREM 3. *The long run probability D of a cache fault in the LRU algorithm is given by*

$$1 - D = \sum_{q=0}^{k-1} [u^q] \int_0^\infty \Phi(t,u)\Psi(t,u)\, e^{-t} dt, \tag{17}$$

where functions Φ and Ψ are

$$\Phi(t,u) = \prod_{i=1}^{m} \left(1 + u(e^{p_i t} - 1)\right) \qquad \text{and} \qquad \Psi(t,u) = \sum_{i=1}^{m} \frac{p_i^2}{1 + u(e^{p_i t} - 1)}. \tag{18}$$

PROOF: The long run probability of a page fault due to a reference to a_i is well defined as the joint probability of events *"a fault occurs"* and *"the referenced item is a_i"*; the page fault probability D follows then by summation of these quantities for $i = 1, \ldots, m$. By symmetry, we only need to consider the case $i = m$, and to simplify notations, we write $a = a_m$, $p = p_m$.

Our line of proof to determine these probabilities is to use suitable shuffle decompositions for languages representative of LRU caching, compute corresponding generating functions by means of Theorem 0, and perform an asymptotic evaluation of those probabilities over *finite* reference sequences.

Shuffle decomposition. Let $\mathcal{B} = \mathcal{A} \setminus \{a\}$. Any word of $\mathcal{A}^*$ can be decomposed according to its occurrences of letter a:

$$\mathcal{A}^* = (\mathcal{B}^* a)^* \, \mathcal{B}^*. \tag{19}$$

This equation only expresses the fact that a word is formed by an alternation of $\mathcal{B}$-blocks ($\mathcal{B}^*$) and of a-letters, so that any $w \in \mathcal{A}^*$ can be written as

$$w = \beta_1 a \beta_2 a \beta_3 \ldots a \beta_{s+1} \qquad \text{with} \qquad \beta_j \in \mathcal{B}^*.$$

Let us assume that the cache initially contains letter a (we know already that this assumption does not affect long run probabilities). Then, the combinatorial condition that determines faults on a-references takes an extremely simple form:

> *An a-letter gives rise to a fault if and only if the preceding $\mathcal{B}$-block β_j contains more than $k-1$ different elements.*

We decompose $\mathcal{B}^*$ as a union of two sets, $\mathcal{B}^* = R + S$ where R is the language formed with words in $\mathcal{B}^*$ having at most $k-1$ different letters, and $S = \mathcal{B}^* \setminus R$. Thus, S is formed of words with at least k different letters and at most $m-1$ different letters, and decomposition (19) can be refined as

$$\mathcal{A}^* = ((R + S)a)^* \, \mathcal{B}^*. \tag{20}$$

Generating functions. Theorem 0 enables us to determine generating functions of the various languages appearing in (20). For a and $\mathcal{B}^*$, we find respectively

$$pz, \quad \frac{1}{1-(1-p)z}. \tag{21}$$

Let $r(z)$ and $s(z)$ be the OGFs of R and S, with $\hat{r}(z)$ and $\hat{s}(z)$ the corresponding EGFs. Employing the notation $a^+ = a \cdot a^*$, we have

$$R = \bigcup_{0 \le |J| < k} \left(a_{j_1}^+ \sqcup\!\sqcup\, a_{j_2}^+ \sqcup\!\sqcup \cdots \sqcup\!\sqcup\, a_{j_r}^+\right) \tag{22}$$

the union being over all sets $J = \{j_1, j_2, \ldots, j_r\}$ such that $m \notin J$ and $1 \le r = |J| < k$. With $\Phi_m(z, u)$ defined by

$$\Phi_m(z, u) = \prod_{i=1}^{m-1} \left(1 + u(e^{p_i z} - 1)\right), \tag{23}$$

we find from Theorem 0 and an argument similar to the one developed for Eqs (8-10):

$$\hat{r}(z) = \sum_{q=0}^{k-1} [u^q]\, \Phi_m(z, u). \tag{24a}$$

From there, we obtain the OGF of R by a Laplace-Borel transform,

$$r(z) = \sum_{q=0}^{k-1} \int_0^\infty [u^q]\left(\Phi_m(zt, u)\right) e^{-t} dt \qquad \text{and} \qquad s(z) = \frac{1}{1-(1-p)z} - r(z). \tag{24b}$$

Probabilities. We now consider the bivariate generating function

$$A(z; v) = \left(\frac{1}{1 - (vr(z) + s(z))pz}\right)\left(\frac{1}{1-(1-p)z}\right), \tag{25}$$

which is constructed as follows: When $v = 1$, form (25) is obtained directly from decomposition (20) by basic translation principles of Theorem 0, so that $A(z; 1) = (1 - z)^{-1}$. More generally, variable v in $A(z; v)$ "marks" occurrences of R-blocks in decomposition (20), and the reader should have no difficulty in convincing herself that the coefficient $[z^n v^t]A(z; v)$ represents the probability of having t R-blocks (i.e. a-hits) in a random word of length n. Thus the quantity

$$\delta_n = [z^n] \left.\frac{\partial}{\partial v} A(z; v)\right|_{v=1} \tag{26}$$

is the expected number of R-blocks in a random word of length n.

Equation (25) and the fact that $A(z; 1) = (1 - z)^{-1}$ permit us to complete the computation of the derivative in (26),

$$\delta(z) \stackrel{\text{def}}{\equiv} \left.\frac{\partial}{\partial v} A(z; v)\right|_{v=1} = \frac{(1-(1-p)z)pz\, r(z)}{(1-z)^2}. \tag{27}$$

From the fact that $R \subset \mathcal{B}^*$, we know that $r(z)$ is analytic for $|z| < (1-p)^{-1}$. Thus, the asymptotic behavior of the coefficient δ_n is fully determined by the behavior of its generating function at the (double) pole $z = 1$: If $\delta(z) \sim c(1 - z)^{-2}$ as $z \to 1$, then $\delta_n \sim cn$ as $n \to \infty$. We thus get

$$\delta_\infty \stackrel{\text{def}}{\equiv} \lim_{n\to\infty} \frac{\delta_n}{n} = \lim_{z\to 1} (1 - z)^2 \delta(z) = p^2 r(1). \tag{28}$$

Quantity δ_∞ in (29) is the long run probability of the event: *"the referenced item is $a = a_m$"* and *"a hit occurs"*. From (23), (24) and the relation $\Phi(z,u) = (1 + u(e^{p_m z} - 1))\Phi_m(z,u)$, we find

$$\delta_\infty = \sum_{q=0}^{k-1} \int_0^\infty [u^q] \left(\Phi(t,u) \frac{p_m^2}{1 + u(e^{p_m t} - 1)} \right) e^{-t} dt. \tag{29}$$

The statement of the theorem now follows by summing quantities obtained from (29) replacing m with $i = 1, 2, \ldots, m$, which corresponds to taking into account all possible references $a_1, a_2, \ldots, a_m$. ■

A symmetric expression that closely resembles (14a) is obtained by expanding the integrand and evaluating resulting integrals.

COROLLARY 4. *The cache fault probability D satisfies*

$$1 - D = \sum_{i=1}^m p_i^2 \sum_{q=0}^{k-1} (-1)^{k-1-q} \binom{m-q-2}{m-k-1} \sum_{\substack{|J|=q \\ i \notin J}} \frac{1}{1 - P_J} \qquad \text{with} \qquad P_J = \sum_{j \in J} p_j. \tag{30}$$

King [1971] gave another form of this probability, but with the same order of complexity, namely $(m^k/k!)$. Due to this complexity, King's numerical data are limited to $m = 9$ and $k \leq 7$. In Section 7, Theorem 5, we shall give brief indications on the possibility of exploiting the result of Theorem 3 instead of these huge combinatorial formulæ.

Formulæ (17) and (30) can be checked against special cases. If $k = 0$ the sum in (17) is empty so that $D = 1$ and there is a page fault at each reference. If $k = 1$, expression (30) gives

$$D = 1 - \sum_{i=1}^m p_i^2,$$

which is the probability that a reference differs from the immediately preceding one. If $k = m$, then $D = 0$ and there is no page fault.

If the probability distribution is uniform, $p_i = 1/m$, then we should have $1 - D = k/m$. To see it, it suffices to check that δ_∞ in (29) is equal to k/m^2. This fact comes from the identity

$$\binom{m-1}{q} \int_0^\infty \left(e^{t/m} - 1 \right)^{m-1} e^{-t} \, dt = 1.$$

Setting $e^{-t/m} \mapsto v$ reduces the integral to an Eulerian Beta integral and the identity follows.

Cold Start Analysis and Transient Behavior. We show here how a simple modification of our previous argument yields the transient behavior of LRU caching. Our earlier analysis has concerned itself with a steady state analysis, and to simplify computations, we have assumed that element a was initially in the cache. This has the effect that the first a-reference behaves like all other references; globally this is equivalent to assuming the cache to be initially in an idealized state that contains all the elements.

A realistic assumption is that the cache is initially empty when the system is started. A corresponding analysis is sometimes called a *cold start* analysis and is of practical relevance [Smith 1982]. We let D_n be the page fault probability in n steps of LRU caching, and (fixing again $a = a_m$, $p = p_m$) δ_n be the expected number of a-hits (non faults) under a sequence of n random references, when the cache is initially empty. (Thus D_n is a sum of the δ_n/n corresponding to a-references for $a = a_1, \ldots, a_m$.)

To determine δ_n, we use a slight modification of decomposition (20), namely

$$\mathcal{A}^* = \left(\epsilon + \mathcal{B}^* a\left(\left(R+S\right)a\right)^*\right)\mathcal{B}^*.$$

The first a is always a fault. Thus the bivariate generating function $A(z;v)$ of $\mathcal{A}^*$ with v marking hits becomes

$$A(z;v) = \left(1 + \frac{pz}{1-(1-p)z}\cdot\frac{1}{1-\frac{pz}{1-(1-p)z}(vr(z)+s(z))}\right)\cdot\frac{1}{1-(1-p)z}.$$

Let $\delta(z) = \frac{\partial}{\partial v}A(z;v)|_{v=1}$. Function $\delta(z)$ is the OGF of δ_n, and a simple computation shows that

$$\delta(z) = \frac{p^2 z^2\, r(z)}{(1-z)^2}.$$

To extract δ_n, we expand the integral expressing $\hat{r}(z)$, next compute $r(z)$ as a Laplace transform $(e^{\alpha z} \mapsto (1-\alpha z)^{-1})$. By partial fraction expansions, the coefficients of $\delta(z)$ can be found.

Corollary 5. *The transient page fault probabilities of LRU caching in n steps, assuming a "cold start" are given by*

$$D_n = D + \frac{E}{n} + F(n)$$

where D is the long run probability of a page fault, and E, $F(n)$ are

$$E = \sum_{i=1}^{m} p_i^2 \sum_{q=0}^{k-1} (-1)^{k-1-q}\binom{m-q-2}{m-k-1}\sum_{\substack{|J|=q\\ i\notin J}}\frac{2-P_J}{(1-P_J)^2}$$

$$F(n) = -\sum_{i=1}^{m} p_i^2 \sum_{q=0}^{k-1} (-1)^{k-1-q}\binom{m-q-2}{m-k-1}\sum_{\substack{|J|=q\\ i\notin J}}\frac{P_J^n}{(1-P_J)^2}$$

with $P_j = \sum_{j\in J} p_j$.

The term E/n represents the amortized effect of a cold start. Quantity $F(n)$ is $O((1-p_{\min})^n)$ with $p_{\min}$ the smallest of the probabilities; it represents a standard Markovian convergence term with exponential decay.

Note on the Independent Reference Model. The Independent Reference Model has been often criticized since actual program paging exhibits localities not captured by the model. Thus if one uses simulation traces to determine both actual page accesses and page faults, one finds that the model based on observed access frequencies provides rather pessimistic estimates. However, Baskett and Rafii [1976] show that, by introducing in the model virtual probabilities that are computed in an appropriate way one can obtain excellent agreement between observed and predicted performance. (The computation scheme constitutes their so-called A0 inversion model.) Therefore, *an actual paging system with* dependent *references can be modeled accurately by the* independent *reference model with modified access probabilities.*

6. Self-Organizing Search

The standard sequential search procedure is well summarized by Knuth [1973, Sec 6.1]: "Begin at the beginning and go on till you find the right key: then stop". In the context where keys, represented by set $\mathcal{A}$, have fixed access probabilities, the optimal arrangement of elements is in order of decreasing probability. Various heuristics have been proposed to handle the situation where access probabilities are not known in advance. Two classical rules are "Transposition" and "Move-To-Front" (MTF):

MTF. When an element in position j is accessed, it is moved to the front of the file. Elements in position $1, 2, \ldots, j-1$ are shifted back by one position.

Formally, the MTF rule is exactly an LRU caching algorithm in which the cache size k is equal to the file size $m = \mathrm{card}(\mathcal{A})$. Page faults disappear and the transition from state K_{t-1} to state K_t is given by rule (16b), where $w_1 w_2 w_3 \ldots$ is the sequence of accessed elements. If at time $t-1$, the file state is $K_t = (c_t^{(1)}, \ldots, c_t^{(m)})$, and the accessed element w_t is in position j, $w_t = c_t^{(j)}$, then the corresponding access cost is taken here to be $j-1$. Thus an element on top of the list has access cost 0. (This convention is adopted to simplify computations.)

Our purpose is only to show that the analysis of MTF can be cast in the framework of shuffles of languages. The theorem that follows is due to McCabe [1965], useful references on the subject being [Knuth 1973], [Rivest 1976], [Bitner 1979], [Gonnet *et al.* 1981].

THEOREM 4 [McCABE 1965]. *The expected cost of a search with the Move-To-Front heuristic applied to a file with access probabilities* $\{p_i\}_{i=1}^m$ *is*

$$E = -\frac{1}{2} + \sum_{1 \le i,j \le m} \frac{p_i p_j}{p_i + p_j}.$$

PROOF. We decompose the cost as $E = f_1 + f_2 + \cdots + f_m$, where f_j is the (long run) cost of an a_j-reference. We evaluate the contribution f_m and set $a = a_m$, $p = p_m$ and $f = f_m$. Our starting point is decomposition (19). We let $B(z;v)$ and $\hat{B}(z;v)$ be the OGF and EGF of $\mathcal{B}^*$ with v marking the number of distinct letters,

$$B(z;v) \stackrel{\text{def}}{\equiv} \sum_{w \in \mathcal{B}^*} \pi[w] v^{d(w)} z^{|w|}, \qquad \hat{B}(z;v) \stackrel{\text{def}}{\equiv} \sum_{w \in \mathcal{B}^*} \pi[w] z^{d(w)} \frac{z^{|w|}}{|w|!},$$

where $d(w)$ is the number of distinct letters in w. By an argument that should now be familiar, we have

$$\hat{B}(z;v) = \prod_{i=1}^{m-1} \big(1 + v(e^{p_i z} - 1)\big). \tag{31}$$

Decomposition (19) suggests to define (compare with Eq (25))

$$A(z;v) = \frac{1}{1 - B(z;v)pz} \cdot \frac{1}{1-(1-p)z} \tag{32}$$

and the reader can again convince herself that the coefficient $[z^n v^c]A(z;v)$ is the probability that the total cost of a-references equals c in a random sequence of length n. Thus

$$f = \lim_{n\to\infty} \frac{1}{n} [z^n] \frac{\partial A(z;v)}{\partial v}\bigg|_{v=1}. \tag{33}$$

The derivative is easily computed,

$$A'_v(z;1) = \frac{pz(1-(1-p)z)}{(1-z)^2} B'_v(z;1).$$

It can be checked that $B(z;v)$ is analytic for $|z| \le 1$ (see also the remarks following this theorem). Considering the double pole $z = 1$, we find

$$f = \lim_{z\to 1}(1-z)^2 A'_v(z;1) = p^2 B'_v(1;1). \tag{34}$$

But from (31) via the Laplace-Borel transform and differentiation, we obtain

$$B'_v(1;1) = \sum_{j=1}^{m-1} \int_0^\infty \left(e^{(1-p)t} - e^{(1-p-p_j)t}\right) e^{-t} dt = \sum_{j=1}^{m-1} \left(\frac{1}{p} - \frac{1}{p+p_j}\right) \tag{35}$$

From there, we find $f = f_m$ under the form

$$f_m = \sum_{j=1}^{m-1} \frac{p_j p_m}{p_j + p_m} = -\frac{p_m}{2} + \sum_{j=1}^{m} \frac{p_j p_m}{p_j + p_m}. \tag{36}$$

The statement of Theorem 4 follows by summing expression (36) with m being replaced by $1, 2, \ldots, m$. ∎

Our line of proof, admittedly not the simplest possible, "explains" the derivation of Theorem 4 that appears in [Knuth 1973, p.403]. In essence Knuth's derivation amounts to operating with an ordinary generating function equivalent to $B(z;v)$ and computed directly by summing over all possible cases. Our proof also yields information regarding the transient behavior of the system. The coefficient $[z^n]A'_v(z;1)$ is the a-cost in n steps, and from observation of the smallest pole, we can deduce Bitner's result: The error term in the convergence to the stationary cost is of the form $O((1 - p_{\min} - p'_{\min})^n)$, where $p_{\min}$ and $p'_{\min}$ are the two smallest probabilities.

7. Some Conclusions

It is clear that our derivations are not "unique", and alternative combinatorial or probabilistic arguments could be (or have been) given for some of our results. Our goal has been to show how addition of the shuffle product to regular languages leads to direct analysis of a natural class of random allocation problems. That approach is of value in more complex situations. For instance, problems around multilevel caching are natural candidates and they are discussed in [Gardy 1989].

Furthermore, with the single exception of self-organizing search, the integral expressions that constitute the natural outcome of these analyses are normally easier to evaluate than the symmetric expressions that we encountered after expanding integrals. Our purpose is not to develop a full theory of numerical evaluation of those integrals, a question which requires further study and is somewhat outside the scope of this paper. However, in order to illustrate the usefulness of integral representations, we offer a brief and informal discussion of two problems, CCP under Zipf's law and LRU caching. Notice that accurate numerical integration algorithms are known, for instance Romberg's acceleration of convergence method [Henrici 1977]: On a well-conditioned function, a few hundred function evaluations will typically guarantee a relative accuracy between 10^{-6} and 10^{-8}.

Zipf's law, a surprising law of Nature, is the probability distribution that assigns to item i the probability c/i. Over a set $\mathcal{A}$ with cardinality m, the normalization constant is $c = 1/H_m$, with H_m a harmonic number. The starting point for CCP is Eq (13b), which we repeat here

$$\mathrm{E}\{C_m\} = \int_0^\infty (1 - \Theta(t))\, dt, \qquad \text{where} \qquad \Theta(t) = \prod_{i=1}^{m} \left(1 - e^{-p_i t}\right) \tag{37}$$

It can be proved that $\Theta(t)$ has a sharp transition from 0 to 1 for t around $m \log^2 m$. More precisely, quantity $F_m(x) = -\log \Theta(xm \log m H_m)$ is such that for fixed x as $m \to \infty$, we have: $F_m(x) \to \infty$ if $x < 1$ and $F_m(x) \to 0$ if $x > 1$. Hence:

COROLLARY 6. *Under a Zipf distribution of parameter m, the expected time of a full coupon collection satisfies*

$$\mathrm{E}\{C_m\} \sim m \log^2 m$$

For instance, the values of $\mathrm{E}\{C_m\}$ when $m = 10, 20, 50, 100, 200, 500, 1000$ are (numbers in brackets represent the corresponding figures for a uniform distribution):

56 (29), 170 (72), 683 (225), 1857 (519), 4873 (1176), 16702 (3396), 41289 (7485).

LRU caching, as well as several other problems discussed in this paper, has solutions expressed as integrals of coefficients of bivariate functions. If $g(u)$ is analytic in $|u| \leq 1$, its Taylor coefficients can be estimated by Cauchy's formula,

$$\begin{aligned} [u^q]g(u) &= \frac{1}{2i\pi} \oint g(u)\, \frac{du}{u^{q+1}} \\ &= \frac{1}{2\pi} \int_0^{2\pi} g(e^{i\theta}) e^{-qi\theta}\, d\theta. \end{aligned} \tag{38}$$

Sums of coefficients can be similarly determined since

$$1 + e^{-i\theta} + e^{-2i\theta} + \cdots + e^{-(k-1)i\theta} = \frac{1 - e^{-ki\theta}}{1 - e^{-i\theta}}. \tag{39}$$

This device can be applied to the various sums that we have encountered (integrating over a smaller circle when necessary). The integrand in the LRU analysis, Eq (17), being analytic for $|u| < 1/(1 - p_{\min})$ with $p_{\min}$ the smallest of the p_i, formulæ (38), (39) can be safely applied.

THEOREM 5. *The page fault probability of LRU caching is expressible as the double integral*

$$1 - D = \frac{1}{2\pi} \int_0^{2\pi} \int_0^\infty \Phi(t, e^{i\theta}) \Psi(t, e^{i\theta}) e^{-t} \frac{1 - e^{-ki\theta}}{1 - e^{-i\theta}}\, d\theta\, dt \tag{40}$$

where $\Phi(t, u) = \prod_{i=1}^m (1 + u(e^{p_i t} - 1))$ *and* $\Psi(t, u) = \sum_{i=1}^m p_i^2/(1 + u(e^{p_i t} - 1))$.

At an abstract level, we roughly estimate that formula (40) provides an analysis of LRU caching that has complexity $O(mk)$: Each evaluation of Φ and Ψ in the integrand has cost $O(m)$. In order to estimate the k-th Taylor coefficient of a function $g(u)$ (or k-th Fourier coefficient of $g(e^{i\theta})$), we expect to perform $O(k)$ evaluations of $g(u)$ since a numerical integration routine should sample enough points on each of the k "waves" of $e^{ki\theta}$. This comparatively low complexity is to be contrasted with a cost of about $\binom{m}{k}$ for the combinatorial sums of Corollary 4, and an even higher cost for a direct Markovian analysis on a chain with $k!\binom{m}{k}$ states. For instance, realistic values of $m = 1000$ and $k = 20$ lead to a Markov chain with about 10^{60} states, which could (in theory!) be solved using time of the order of 10^{180} and space 10^{60}. The cost of evaluating the combinatorial sums is still about 10^{40}. In contrast, about $10^7/10^8$ elementary function evaluations should suffice to evaluate the cache fault probabilities in this case.

As a final note, shuffles of regular languages have been found useful in a few other places in the analysis of algorithms. Thimonier's dissertation [1988] provides a review with several other applications. The problem of analyzing the evolution of two stacks in a common memory area [Knuth 1968, Ex 2.2.2.13] has been solved by one of us, using shuffles of one-dimensional random walk languages [Flajolet 1986]. An interesting approach to the evaluation of concurrency control algorithms and mutual exclusion is developed by Françon in [Françon 1986].

Acknowledgements. The first author (by alphabetical order) would like to express his gratitude to Donald Knuth for an invited visit at Stanford during which his work was done for a large part under support of the National Science Foundation grant CCR-8610181 and Office of Naval Research grant N-00014-87-K-0502. D.E.K. made several useful suggestions, notably a simplification in our original analysis of LRU caching.

References

O. I. Aven, L. B. Boguslavsky and Y. A. Kogan [1976] Some results on distribution–free analysis of paging algorithms, *IEEE Trans. on Computers* **C-25**, 737-745 (1976).

F. Baskett and A. Rafii [1976], The A0 inversion model of program paging behavior, Stanford University Report CS-76-579 (1976).

L. Belady [1966], A study of replacement algorithms for a virtual storage computer, *IBM Systems J.* **4**, 297-305 (1966).

J. R. Bitner [1979], Heuristics that dynamically organize data structures, *SIAM J. on Computing* **8**, 82-110 (1979).

R. P. Brent [1981], An improved Monte Carlo factorization, *BIT* **20**, 176-184 (1981).

E. G. Coffman and P. J. Denning [1973], *Operating Systems Theory*, Prentice Hall (1973).

L. Comtet [1974], *Advanced Combinatorics.* Reidel, Dordrecht (1974).

R. Fagin [1977], Asymptotic miss ratios over independent references, *J. Computer and System Sc.* **14**, 222-250 (1977).

R. Fagin and T. G. Price [1978]. Efficient calculation of expected miss ratios in the independent reference model, *SIAM J. on Computing* **7**, 288-297 (1978).

W. Feller [1968], *An Introduction to Probability Theory and its Applications,* J. Wiley (1968).

P. Flajolet [1986], The evolution of two stacks in bounded space and random walks in a triangle, in Proc. MFCS, Bratislava, *Lect. Notes in Comp. Science* **233**, 325-340 (1986)

P. Flajolet [1988], Mathematical analysis of algorithms and data structures, in *Current Trends in Theoretical Computer Science*, E. Börger Ed., Computer Science Press, Rockville, 225-304 (1988).

P. Flajolet, D. Gardy and L. Thimonier [1988], Probabilistic languages and random allocations, in *Automata, Languages and Programming* (Proceedings of the 15th ICALP), *Lect. Notes in Comp. Science* **317**, 239–253 (1988).

J. FRANÇON [1986], Une approche quantitative de l'exclusion mutuelle, *RAIRO Theoretical Informatics and Applications* **20**, 275-289 (1986).

D. R. FUCHS AND D. E. KNUTH [1985], Optimal prepaging and font caching, *ACM Trans. on Programming Lang. Systems* **7**, 62-79 (1985).

D. GARDY [1989], Bases de données, allocations aléatoires: quelques analyses de performances, Doctorate in Sciences, Université de Paris-Sud (June 1989).

E. GELENBE [1973], A unified approach to the evaluation of a class of random replacement algorithms, *IEEE Trans. Computers* **C-22**, 611-618 (1973).

G. H. GONNET, J. I. MUNRO AND H. SUWANDA [1981], Exegesis of self-organizing linear search, *SIAM J. on Computing* **10**, 613-637 (1981).

I. GOULDEN AND D. JACKSON [1983], *Combinatorial Enumerations.* J. Wiley, New-York (1983).

D. GREENE [1983], Labelled Formal Languages and Their Uses, PhD Thesis, Stanford University Report STAN-CS-83-982 (1983), 148p.

P. HENRICI [1977], *Applied and Computational Complex Analysis, Vol 2,* Wiley (1977).

W. C. KING [1972], Analysis of paging algorithms, in *Proc. IFIP 1971 Congress, Ljubljana,* North-Holland, 485-490 (1972).

M. S. KLAMKIN AND D. J. NEWMAN [1967], Extensions of the birthday surprise, *J. of Combinatorial Theory* **3**, 279-282 (1967).

D. E. KNUTH [1968], *The Art of Computer Programming, Vol 1: Fundamental Algorithms.* Addison-Wesley, Reading (1968).

D. E. KNUTH [1969], *The Art of Computer Programming, Vol 2: Seminumerical Algorithms.* Addison-Wesley, Reading (1969).

D. E. KNUTH [1973], *The Art of Computer Programming, Vol 3: Sorting and Searching.* Addison-Wesley, Reading (1973).

D. E. KNUTH [1985], An analysis of optimum caching, *J. of Algorithms* **6**, 181-199 (1985).

D. E. KNUTH AND G. S. RAO [1975], Activity in an interleaved memory, *IEEE Trans. on Computers* **C-24**, 943-944 (1975).

D. E. KNUTH AND A. SCHÖNHAGE [1978], The expected linearity of a simple equivalence algorithm, *Theoretical Computer Science* **6**, 281-315 (1978).

M. LOTHAIRE [1983], *Combinatorics on Words,* Encyclopedia of Mathematics and Its Applications, Vol 17, Academic Press (1983).

S. MASE [1990], The surname problem—a variation of the birthday problem with unequal probabilities of occurrence, Technical Report No. 264, Statistical Research Group, Hiroshima University, 17 pages (March 1990).

J. MCCABE [1965], On serial files with relocatable records, *Operations Res.* **12**, 609-618 (1965).

H. B. NATH [1973], Waiting time in the coupon collector's problem, *Austral. J. Statist.* **15** (2), 132–135 (1973)

R. RIVEST [1976], On self-organizing sequential search heuristics, *Comm. ACM* **19**, 63-67 (1976).

A. SALOMAA AND M. SOITTOLA [1978], *Automata Theoretic Aspects of Formal Power series,* Springer Verlag (1978).

A. J. SMITH [1982], Cache memories, *ACM Computing Surveys* **14**, 473-530 (1982)

R. P. Stanley [1986], *Enumerative Combinatorics,* Wadsworth and Brooks/Cole, Monterey (1986).

L. Thimonier [1988], Fonctions génératrices et mots aléatoires, Doctorate in Sciences, Université Paris-Sud (January 1988).

C. S. Wetherell [1980], Probabilistic languages, a review and some open questions, *ACM Computing Surveys* **12**, 361-379 (1980).