AVERAGE-CASE PERFORMANCE
OF THE APRIORI ALGORITHM∗

PAUL W. PURDOM†, DIRK VAN GUCHT†, AND DENNIS P. GROTH‡

Abstract. The failure rate of the Apriori Algorithm is studied analytically for the case of random shoppers. The time needed by the Apriori Algorithm is determined by the number of item sets that are output (successes: item sets that occur in at least \( k \) baskets) and the number of item sets that are counted but not output (failures: item sets where all subsets of the item set occur in at least \( k \) baskets but the full set occurs in less than \( k \) baskets). The number of successes is a property of the data; no algorithm that is required to output each success can avoid doing work associated with the successes. The number of failures is a property of both the algorithm and the data.

We find that under a wide range of conditions the performance of the Apriori Algorithm is almost as bad as is permitted under sophisticated worst-case analyses. In particular, there is usually a bad level with two properties: (1) it is the level where nearly all of the work is done, and (2) nearly all item sets counted are failures. Let \( l \) be the level with the most successes, and let the number of successes on level \( l \) be approximately \( \binom{m}{l} \) for some \( m \). Then, typically, the Apriori Algorithm has total output proportional to approximately \( \binom{m}{l} \) and total work proportional to approximately \( m/(l + 1) \).

The analytical results for random shoppers are compared against measurements for three data sets. These data sets are more like the usual applications of the algorithm. In particular, the buying patterns of the various shoppers are highly correlated. For most thresholds, these data sets also have a bad level. Thus, under most conditions nearly all of the work done by the Apriori Algorithm consists in counting item sets that fail.

Key words. data mining, algorithm analysis, Apriori Algorithm

AMS subject classification. 68W40

DOI. 10.1137/S0097539703422881

1. Introduction. The Apriori Algorithm [2, 3, 6, 16] solves the frequent item sets problem, which is at the core of various algorithms for data mining problems. The best known such problem is the problem of finding the association rules that hold in a basket-items relation [2, 3, 16, 20]. Other data mining problems based on the Apriori Algorithm are discussed in [6, 14, 16, 18, 21, 22].

The Apriori Algorithm analyzes a data set of baskets, each containing a set of items, to determine which combination of items occur together frequently. Consider a store with \( |I| \) items where \( b \) shoppers each have a single basket. Each shopper selects a set of items for their basket. The input to the Apriori Algorithm is a list giving the contents of each basket. For a fixed threshold \( k \), the algorithm outputs a list of those sets of items that are frequent; that is, they are contained in at least \( k \) of the \( b \) baskets.

The Apriori Algorithm is a level-wise algorithm. It considers sets of items in order of their size: first sets of size one are tested to see whether they are contained in \( k \) baskets, then sets of size two, etc. On each level, the algorithm knows the result.

∗Received by the editors February 14, 2003; accepted for publication (in revised form) January 26, 2004; published electronically July 29, 2004. The work of the second and third authors was supported by NSF grant IIS-0082407.
†Computer Science Department, Indiana University, Bloomington, IN 47405-4101 (pwp@cs.indiana.edu, vgucht@cs.indiana.edu).
‡School of Informatics, Indiana University, Bloomington, IN 47405-4101 (dgroth@indiana.edu).
of the previous level. The key idea of the Apriori Algorithm is that, on level $l$, a set is tested if and only if all its subsets of size $l - 1$ are frequent. An item set satisfying this property is called a candidate (at level $l$). It is a success (at level $l$) if it is contained in at least $k$ baskets, and it is a failure otherwise.

This paper considers only the Apriori Algorithm. Many of its performance characteristics come from the fact that it outputs every frequent item set. There are several interesting algorithms for the frequent item set problem that output only the maximal frequent item sets, since every subset of a frequent item set is also frequent [1, 4, 12]. Also, there is an algorithm complementary to the Apriori Algorithm that finds the infrequent item sets, starting with the set of all items and working downward in set size.

This paper considers the expected amount of work that the Apriori Algorithm does when the shoppers shop at random. Specifically, the probability that a shopper buys an item is $p$, independent of all other shoppers and all other items. This is the same probability model that has previously been used to estimate the probability that a set is frequent [3, 17, 21]. The main advantage of using a parameterized probability model is that we can study the performance of the algorithm under a wide range of conditions. While the later sections contain many mathematical details, the main conclusions are fairly simple. For most values of the parameters, random data result in a high success rate (essentially 1) for small values of $l$, with a sudden switch to a low success rate (essentially 0) for larger values of $l$. When $l$ is below the level of the switch, the number of candidates is just below $\binom{|I|}{l}$ and almost all of the candidates are successes. When $l$ is above the level of the switch, almost all candidates are failures, and the number of candidates decreases rapidly. The level for the switch depends on the parameters. When $l$ is much less than $|I|$, nearly all the work of the algorithm is done testing candidates from the level with the most successes. The algorithm has a bad level where (1) most of the work is done and (2) there are a lot of candidates, but few successes. For a few thresholds, two levels in a row are bad.

The main disadvantage of using a simple probability model is that most applications of the Apriori Algorithm involve data sets with complex correlations, and such data sets are difficult to model mathematically. Therefore, we also measured the success, candidate, and failure rates of the performance of the Apriori Algorithm on three data sets: (1) synthetic data generated by the generator from the IBM Quest Research Group [13]; (2) U.S. Census data, using Public Use Microdata Samples (PUMS) (the same sample that was used by [5, 6] and processed in the same way); and (3) a Web data set [24]. In one important way, the results with these data sets were similar to that of random shoppers. For most thresholds, the experimental data show the existence of a single bad level. This bad level came just after the level with the most successes, it had a low success rate, and it consumed most of the work that the algorithm did. On the other hand, with the experimental data there are also many failures below the bad level, so that the number of candidates is much less than $\binom{|I|}{l}$.

2. The Apriori Algorithm. Let $J_l$ be a set of $l \geq 1$ items where the items are selected from the set $I$. For a given $J_l$, define $J_l^{-h}$ to be the set obtained from $J_l$ by omitting element $h$, where $h$ is an element of $J_l$. The key idea of the Apriori Algorithm is that the set $J_l$ cannot possibly have $k$ occurrences unless, for each $h$ in $J_l$, the set $J_l^{-h}$ has $k$ occurrences. Since the algorithm considers possible sets in order of their size, it has already gathered the information about all the sets of size $l - 1$ before it considers sets of size $l$. 
Apriori Algorithm.

Step 1. For \( l \) from 1 to \( |I| \) do

Step 2. For each set \( J_l \) such that for each \( h \in J_l \) the set \( J_l^{-h} \) occurs in at least \( k \) baskets do

Step 3. Examine the data to determine whether the set \( J_l \) occurs in at least \( k \) baskets. Remember those cases where the answer is “yes.”

Step 2 of the algorithm generates each set \( J_l \) such that \( J_l^{-h} \) occurs at least \( k \) times (for each \( h \) in \( J_l \)). The sets that are generated are called candidates. For those sets that are candidates, Step 3 examines the data (basket contents) to determine which set of items occurs in at least \( k \) baskets. This counting and comparing with the threshold is called the frequency test. A candidate that passes the frequency test is called a success (or a frequent item set). A set that is subjected to the frequency test but fails is called a failure. For algorithms that verify each success by doing the frequency test, the main place for improvement is to reduce the number of failures.

For typical data sets, a careful implementation of the Apriori Algorithm will have it spending most of its time accessing the data base (counting the occurrences of the various candidates). The implementation should exit the loop in Step 1 early if there are no “yes” answers for some value of \( l \). On level \( l \), it should consider only those sets that are formed from sets that passed the frequency test on level \( l - 1 \). In addition, no set of size \( l \) should be generated more than once. The sets can be generated by assigning an order to the items and extending each set \( S \) on level \( l - 1 \) only with items that are greater than the largest item in \( S \). Assuming unit time for hash table look-ups (for looking up various subsets of the extended \( S \)), the algorithm can do the work for a single candidate set on level \( l \) in time bounded by a constant times \( l + 1 \). See [2] for further discussion of the techniques used in good implementations.

Although the Apriori Algorithm uses explicit counting to verify that an item set is frequent, this is not always logically necessary. Thus, if item \( A \) occurs in \( b_A \) baskets and item \( B \) occurs in \( b_B \) baskets, then \( A \) and \( B \) must occur together in at least \( b_A + b_B - b \) baskets [11]. Similar ideas are explored in [7].

3. Best and worst cases. We use the number of candidates as a proxy for the amount of computing that the Apriori Algorithm does. Let \( N_S \) be the number of item sets that are successes. Let \( N_F \) be the number of failures (candidates that are not successes). The total work is proportional to \( N_S + N_F \). The term \( N_S \) represents work that must be done by any algorithm that outputs every frequent item set, and it is a property of the data, not of the algorithm. On the other hand, \( N_F \) represents work that we might hope to avoid.

The worst-case output is exponentially larger than the input (every set might be frequent), so the worst-case time needed can be exponential in the size of the input. A closely related problem that is NP-complete is to determine whether or not there are any sets of size \( l \) that occur \( k \) times; this is NP-complete because the balanced complete bipartite subgraph problem [9] reduces to it. The appendix has the details of proofs for this and many other statements.

For algorithms that must output every frequent item set, the ratio \( N_S/(N_S + N_F) \) is the natural measure of algorithm efficiency. When every item set is frequent, \( N_S \) is \( 2^{|I|} \) and \( N_F = 0 \), so the algorithm is efficient. The worst ratio of work to output occurs when no (nonempty) item set is frequent, in this case \( N_S = 1 \), \( N_F = |I| \). (Most versions of the algorithm avoid outputting the empty set, leading to values of \( N_S \) one less than above.)

Intermediate amounts of output are more interesting. In such cases, one can have
a lot of output and also a high failure rate. In this regard, the worst-case bounds of Geerts, Goethals, and Van den Bussche [10] are interesting. For their Theorem 1, they write the number of successes on level $l$ as a binomial coefficient with $l$ on the bottom in which the top index is as large as possible subject to the binomial coefficient being no larger than the number of success. If the difference between the number of successes and the binomial coefficient is positive, then they repeat the process with the difference. Thus, they write the number of successes as a sum of binomial terms. They show that the result of increasing each bottom index to $l + 1$ gives an upper bound on the number of candidates for level $l + 1$. This bound is exact for some data sets, including one that results in no failures on level $l$. Experiments show that in many cases this upper bound on candidates is close to the exact value for level $l + 1$. The same idea can be used to predict the number of successes on later levels, but it often works less well for those levels.

The following argument also limits the number of candidates on level $l + 1$. Consider the graph where each candidate on level $l + 1$ (each $J_{l+1}$) is connected to each of its associated frequent items sets ($J_{l+1}$). Thus, there are $[N_S(l+1) + N_F(l+1)](l + 1)$ arcs. Each of the $N_S(l)$ frequent item sets on level $l$ is connected to at most $|I| - l$ candidates on level $l + 1$. Indeed, if we use $|I|$ to be the number of items that occur among the frequent item sets of level $l$, there are at most $|I| - l$ connections from a single level $l$ frequent item set. Thus, we have

\begin{equation}
[N_S(l + 1) + N_F(l + 1)](l + 1) \leq N_S(l)(|I| - l),
\end{equation}

\begin{equation}
[N_S(l + 1) + N_F(l + 1)] \leq \left(\frac{|I| - l}{l + 1}\right) N_S(l).
\end{equation}

This gives

\begin{equation}
\sum_{0 \leq i \leq |I| - 1} \left(\frac{|I| - l}{l + 1}\right) N_S(l)
\end{equation}

as an upper bound on the amount of work. In many cases, $N_S(l)$ increases rapidly for small $l$ and then suddenly drops rapidly to zero. In such cases, the largest term in this sum is a good approximation to its total value.

We see that the Apriori Algorithm is rather efficient for algorithms that output every frequent item set. The total work is never more than a constant plus a factor $|I|$ times larger than $N_S$, the least amount of work that could possibly be done by any algorithm in the class. When most of the work is concentrated on level $l + 1$, the amount of work is better than this product by a factor of $l + 1$.

4. **Average case.** We now start an exact computation of the average-case performance of the Apriori Algorithm for the case when the baskets are filled at random. That is, each basket $j$ has item $i$ with probability $p$ independent of what happens for other baskets and other items. We eventually show that for most values of the parameters, the average performance is not significantly better than that suggested by the worst-case analysis (equation (3)).

Let $S_l$ be the probability that the set consisting of items 1 to $l$ is a frequent item set, and let $F_l$ be the probability that the same set is a candidate but fails to be a frequent item set. Since each basket is filled randomly, any other set of $l$ items has
the same probability of success and failure. The expected number of successes is

\[ \sum_{1 \leq l \leq |I|} \binom{|I|}{l} S_l, \]

and the expected number of failures is

\[ \sum_{1 \leq l \leq |I|} \binom{|I|}{l} F_l. \]

The number of item sets for which the basket data is examined is

\[ \sum_{1 \leq l \leq |I|} \binom{|I|}{l} (S_l + F_l). \]

Under the above assumptions, the running time is bounded by a constant times

\[ \sum_{1 \leq l \leq |I|} (l + 1) \binom{|I|}{l} (S_l + F_l). \]

Define the following conditions with respect to a single basket:

- \( M_0 \): the basket has all the items 1 to \( l \), and
- \( M_h \) (1 \( \leq h \leq l \)): the basket has all items from 1 to \( l \) except that it does not have item \( h \).

These conditions are disjoint; each basket obeys at most one of the conditions \( M_h \), \( 0 \leq h \leq l \).

The probability that a randomly filled basket obeys condition \( M_0 \) is

\[ P(l) = p^l \]

(when \( l \leq |I| \)). The probability that a randomly filled basket obeys condition \( M_h \) (for any \( h \) in the range 1 to \( l \)) is

\[ Q(l) = p^{l-1}(1-p). \]

Note that

\[ P(l - 1) = P(l) + Q(l). \]

The probability that at least \( k \) baskets obey condition \( M_0 \) is

\[ S_l = \sum_{j \geq k} \binom{b}{j} [P(l)]^j [1 - P(l)]^{b-j} = 1 - \sum_{j < k} \binom{b}{j} [P(l)]^j [1 - P(l)]^{b-j}. \]

The probability that \( j_0 \) baskets obey condition \( M_0 \), \( j_1 \) baskets obey condition \( M_1 \), \( \ldots, j_l \) baskets obey condition \( M_l \), and the remaining \( b - j_0 - \cdots - j_l \) baskets do not obey any of the conditions is

\[ \binom{b}{j_0, \ldots, j_l, b-j_0-\cdots-j_l} [P(l)]^{j_0} [Q(l)]^{j_1+\cdots+j_l} [1 - P(l) - lQ(l)]^{b-j_0-\cdots-j_l}. \]
where the multinomial coefficient is the number of ways to arrange \( b \) distinct baskets into \( l + 1 \) sets, where set 0 has \( j_0 \) baskets, \ldots, set \( l \) has \( j_l \) baskets, and \( b - j_0 - \cdots - j_l \) baskets are not in any of the \( l + 1 \) sets.

The item set \( \{1, \ldots, l\} \) is a candidate if and only if for each \( h \) in the range \( 1 \leq h \leq l \) we have the number of baskets satisfying condition \( M_h \) plus the number of baskets satisfying condition \( M_h \) totaling at least \( k \). Thus, item set \( \{1, \ldots, l\} \) is a candidate in just those cases where the conditions

\[
 j_0 + j_1 \geq k, \ j_0 + j_2 \geq k, \ \ldots, \ j_0 + j_l \geq k
\]

are all true. Thus, the probability that the set \( \{1, \ldots, l\} \) is a candidate is the above probability (12) summed over those cases that satisfy the conditions (13),

\[
 C_l = \sum_{j_0 \leq k} \binom{b}{j_0, \ldots, j_l, b - j_0 - \cdots - j_l} \times [P(l)]^{j_0} [Q(l)]^{j_1 + \cdots + j_l} (1 - P(l) - lQ(l))^{b - j_0 - \cdots - j_l}. 
\]

Since the set \( \{1, \ldots, l\} \) either does or does not occur in at least \( k \) baskets, the probability that the item set \( \{1, \ldots, l\} \) is a candidate but not a frequent item set is

\[
 F_l = C_l - S_l = \sum_{j_0 < k} \binom{b}{j_0, \ldots, j_l, b - j_0 - \cdots - j_l} \times [P(l)]^{j_0} [Q(l)]^{j_1 + \cdots + j_l} (1 - P(l) - lQ(l))^{b - j_0 - \cdots - j_l}. 
\]

4.1. Efficient computation of \( F_l \). The number of arithmetic operations needed to compute \( S \) for fixed \( b, l, \) and \( p \) (using the right part of (11)) is \( O(k) \). Furthermore, the number of operations for fixed \( l \) and \( p \) and for all \( k \) is only \( O(b) \).

The number of operations needed to compute \( F \) by direct application of (15) is \( O(kb^l) \). However, using the recurrence equations below, \( F \) can be computed in time that is independent of \( k \) and polynomial in \( b \) and \( l \).

Write (15) as

\[
 F_l = \sum_{j_0 < k} \binom{b}{j_0} [P(l)]^{j_0} R_{k - j_0}(b - j_0, l, l), 
\]

where

\[
 R_k(b, l, m, n) = \sum_{j_1 \geq k} \binom{b}{j_1, \ldots, j_l, b - j_1 - \cdots - j_l} \times [Q(m)]^{j_1 + \cdots + j_l} (1 - P(m) - nQ(m))^{b - j_1 - \cdots - j_l}. 
\]

By considering the sum over \( j_l \) (represented by \( j \) in the sum below) separately, we have

\[
 R_k(b, l, m, n) = \sum_{j \geq k} \binom{b}{j} [Q(m)]^j R_k(b - j, l - 1, m, n), 
\]
With these equations, a particular $R_k(b, l, m, n)$ can be computed from the various $R_k(c, l - 1, m, n)$, where $k \leq c \leq b$, in $O(b)$ operations. To compute $R$ by repeated application of (19), we need $l$ levels with $O(b)$ $R$’s per level. This leads to time $O(b l^2)$ to compute a set of $R$. The time to compute $F_i$ is dominated by the time needed to compute the $R$’s, leading to time $O(b l^2)$ to compute a particular $F$. For fixed $p$ and $b$ and for all $k$, $F_i$ can also be computed in time $O(b l^2)$.

5. Approximations.

5.1. Chernoff bounds. The sums for $S_i$ and $F_i$ are incomplete binomial sums. They do not have closed forms (implied by [15]), but, as we show below, Chernoff techniques [8] lead to useful approximations. For

\begin{equation}
R_k(b, 0, m, n) = [1 - P(m) - nQ(m)]^b.
\end{equation}

and for some fixed $k$ in the range $0 \leq k \leq n$, we have

\begin{equation}
\sum_{0 \leq i \leq k} a_i = \sum_{0 \leq i \leq n} a_i L(i) \quad \text{and} \quad \sum_{k \leq i \leq n} a_i = \sum_{0 \leq i \leq n} a_i U(i).
\end{equation}

In addition, when each $a_i \geq 0$, replacing $L(i)$ (or $U(i)$) with a pointwise upper bound gives an upper bound on the sum. Chernoff [8] noticed that useful bounds for partial binomial sums result when one uses

\begin{equation}
L(i) = x^{-k+i} \quad \text{with } x \leq 1 \quad \text{and} \quad U(i) = x^{-k+i} \quad \text{with } x \geq 1
\end{equation}

and then chooses the $x$ that gives the smallest upper bound.

The Chernoff bound for $S$ is

\begin{equation}
S_i \leq x^{-k} \sum_j \binom{b}{j} [xP(l)]^j [1 - P(l)]^{b-j} = x^{-k} [1 + (x-1)P(l)]^b
\end{equation}

for any $x \geq 1$.

A Chernoff bound for $R$ is

\begin{equation}
R_k(b, l, m, n) \leq x^{-kl} \sum_{j_1,\ldots,j_l} \binom{b}{j_1,\ldots,j_l, b - j_1 - \cdots - j_l}
\times [xQ(m)]^{j_1+\cdots+j_l} [1 - P(m) - nQ(m)]^{b-j_1-\cdots-j_l}
\end{equation}

\begin{equation}
\leq x^{-kl} [1 - P(m) - (n - lx)Q(m)]^b
\end{equation}

for any $x \geq 1$.

Using this Chernoff bound for $R$ leads to the following Chernoff bound for $F$:

\begin{equation}
F_l \leq y^{-k+1} \sum_j \binom{b}{j} [yP(l)]^j x^{-(k-j)l} [1 - P(l) + l(x-1)Q(l)]^{b-j}
\end{equation}

\begin{equation}
\leq x^{-kl} y^{-k+1} [1 + (x'y - 1)P(l) + l(x-1)Q(l)]^b
\end{equation}

for any $x \geq 1$ and any $y \leq 1$. 

with boundary condition

\begin{equation}
R_k(b, l, m, n) = \sum_{i=1}^{n} \binom{b}{i} [xP(l)]^i [1 - P(l)]^{b-i} = \sum_{i=1}^{n} \binom{b}{i} x^i [1 - P(l)]^{b-i}
\end{equation}

AVERAGE-CASE PERFORMANCE OF THE APRIORI ALGORITHM 1229
5.2. Regions and boundaries for $S_l$. The optimum $x$ for (23) is either on the boundary ($x = 1$) or when the derivative with respect to $x$ is equal to zero. Letting $x_*$ be the $x$ value that gives a derivative of zero, we have

$$x_* = \frac{k[1 - P(l)]}{(b - k)P(l)}.$$  

(28)

When $x_* \geq 1$, it is the optimum $x$ for (23). Otherwise ($x_* < 1$) the optimum $x$ is 1. In addition, we check whether $x_*$ is strictly within the range ($x_* > 1$). This is the case when

$$k > bP(l).$$ 

(29)

This completes the first stage of finding the Chernoff approximation to $S_l$. In the next two sections we determine just how small the Chernoff bound is as a function of the parameters ($b$, $k$, $l$, and $p$). We show that the bound on $S_l$ is an exponential function of the negative of the square of the distance $\alpha_1$ (with $\alpha_1 = k/b - P(l)$) inside the boundary (29). As we vary $k$, $S_l$ is extremely small inside the region ($k > bP(l)$), except near the boundary ($bP(l)$). Next we show that $S_l$ is close to 1 once we go on the other side of the boundary; the difference between $S_l$ and 1 is an exponential function of the negative of the square of the distance $\alpha_2$ ($\alpha_2 = (P(l) - k/b) + 1/b = -\alpha_1 + 1/b$) from the boundary. Thus, knowing whether the optimizing $x$ is strictly within range or not gives us the most basic information about $S_l$ (whether it is small ($x_* > 1$) or large ($x_* < 1$)).

5.2.1. Upper bound on $S_l$. We now give an upper bound on $S_l$ when $k > bP(l)$ to show that it is near 0. In the next section we give a lower bound when $k > bP(l)$ to show that in that case it is near 1.

By plugging the $x_*$ value from (28) into the bound from (23) we obtain

$$S_l \leq \left(\frac{P(l)}{k}\right)^k \left(\frac{1 - P(l)}{b - k}\right)^{b-k} b^b$$  

so long as $x_* \geq 1$. By (29) the condition $x_* > 1$ is equivalent to $k > bP(l)$, so we will define $\alpha_1$ by

$$k = b[P(l) + \alpha_1].$$ 

(31)

When $k$ is greater than $bP(l)$, $S_l$ goes to zero rapidly. In particular

$$S_l \leq e^{-b\alpha_1^2/[2P(l)(1-P(l))] + O(b\alpha_1^2[1-P(l)]^{-2})}$$  

(32)

when $\alpha_1 > 0$.

5.2.2. Lower bound on $S_l$. To obtain a lower bound on $S_l$ when it is near 1, start with the right part of (11). Shift the relation between $k$ and $\alpha_1$ by 1 so that $\alpha_2$ is defined implicitly by

$$k = b[P(l) - \alpha_2] + 1.$$ 

(33)

We can now modify the derivation of (32) (with $x_* < 1$) to obtain

$$S_l \geq 1 - e^{-b\alpha_2^2/[2P(l)(1-P(l))] + O(b\alpha_2^2P(l)^{-2})}$$  

(34)

when $\alpha_2 > 0$ (the dominant $O$ term here is different from what it was for (32)).
5.3. Regions and boundaries for $F_l$. To find the optimum value for $x$ and $y$ in (27) we start by taking derivatives of the bound with respect to $x$ and $y$, setting each result to zero, and solving for $x$ and $y$. We want the $x_*$ that satisfies

$$
(b - k)P(l)x_*^l y + (b - kl)Q(l)x_* - k[1 - P(l) - lQ(l)] = 0.
$$

We want the $y_*$ that satisfies

$$
(b - k + 1)P(l)x_*^l y_* - (k - 1)[1 - P(l) + lQ(l)(x_* - 1)] = 0.
$$

When considering whether the optimum $x$ and $y$ are strictly within range ($x_*, y_* > 1$, $y_* < 1$) or on the boundary, there are four cases to investigate.

Region 1. Equation (35) with $y = 1$, $x_* > 1$.
Region 2. Equation (36) with $x = 1$, $y_* < 1$.
Region 3. Equations (35) and (36), $x_* > 1$, $y_* < 1$.
Region 4. $x = 1$, $y = 1$.

In (35) and (36), $x_*$ is associated with the effectiveness of the candidacy test (inequality (25)), and $y_*$ is associated with the probability of a set failing the frequency test (inequality (27)).

The main regions of interest are Region 1, where we will show that $F_l$ is small because the candidacy test fails with high probability; Region 2, where $F_l$ is small because $S_l$ is near 1 ($F_l$ can never be larger than 1 $- S_l$ since failure requires not only passing the candidacy test but also failing the frequency test); and Region 4, where $F_l$ has the trivial bound of 1. In section 5.3.3 we show that the set of parameter values that satisfy the conditions for Region 3 includes the intersection of Regions 1 and 2. Also Region 3 has no values outside of the union of Regions 1 and 2.

When the optimum value for at least one of $x$ and $y$ is strictly within range (not equal to 1) then the bound for $F_l$ is smaller. It will be shown in section 5.4 that the bound on $F_l$ is an exponential function of the square of the distance (basically the difference between $k/b$ and $P(l)$ or $P(l - 1)$; see section 5.4 for details) of $x_*$ or $y_*$ from the boundary, so $F_l$ rapidly becomes extremely small as $x_*$ or $y_*$ moves away from the boundary.

5.3.1. Region 1. When $x_* > 1$, we will show in section 5.4.1 that the candidacy test fails with high probability. To find when this occurs, notice that (35) is satisfied by $x = 1$, $y = 1$ when

$$
k = b[P(l) + Q(l)] = bP(l - 1).
$$

As $b$ decreases, $x_*$ increases. This implies that, for $y = 1$, $x_* > 1$ when

$$
k > bP(l - 1).
$$

5.3.2. Region 2. When $y_* < 1$, we will show in section 5.4.2 that the frequency test succeeds with high probability. Consequently, since $F_l$ can be no larger than $1 - S_l$, $F_l$ is near 0 in this case.

When $x = 1$, the solution to (36) is

$$
y_* = \frac{(k - 1)[1 - P(l)]}{(b - k + 1)P(l)}.
$$

This results in $y_* < 1$ when

$$
k < bP(l) + 1.
$$
For most parameter values, the regions (permitted values of \( k \)) of (38) and (40) do not overlap. However, subtracting the right side of (40) from the right side of (38), we find that they do overlap when

\[ bQ(l) < 1. \] (41)

This happens both when \( p^{l-1} \) is small (\( p^{l-1} \leq 1/b \) is small enough) and also when \( 1 - p \) is small (\( 1 - p \leq 1/b \) is small enough). When (41) is true for all \( l \), the Apriori Algorithm has no bad level. In this case, \( S_l \) is small for every \( l \). Conditions where the Apriori Algorithm does have a bad level (cases where \( S_l \) is near 1) are discussed in section 5.4.4.

5.3.3. Region 3. To find values for the parameters such that \( x^* > 1 \) and \( y^* < 1 \) we need to satisfy (35) and (36) simultaneously. This results in the values

\[ x^* = \frac{1 - P(l) - lQ(l)}{(b - k - l + 1)Q(l)}, \] (42)

\[ y^* = (k - 1) \left( \frac{(b - k - l + 1)Q(l)}{1 - P(l) - lQ(l)} \right)^{l-1} \frac{Q(l)}{P(l)}. \] (43)

We have \( x^* > 1 \) when

\[ k + l - 1 < b < k + l - 1 + \frac{1 - P(l) - lQ(l)}{Q(l)}. \] (44)

The upper and lower limits are the same when \( l = 1 \), so the range is empty in that case.

All solutions to (44) are in the union of Regions 1 (inequality (38)) and 2 (inequality (40)). The smallest \( k \) that satisfies (38) is \( k \) just above \( bP(l - 1) \). This value for \( k \) satisfies (44) when

\[ b < \frac{1}{Q(l)}. \] (45)

Inequality (45) is true under the same conditions that (41) is true. Thus, (44) is satisfied by \( k \) values outside of Region 1 only when Regions 1 and 2 overlap. Since Region 1 gives a lower limit on \( k \) and Region 2 gives an upper limit, when Regions 1 and 2 overlap, their union includes all \( k \) values.

For \( k = 1 \), (43) implies that \( y = 0 \), which is less than 1. For \( l = 1 \), (43) has no solutions. For \( k \geq 2 \) and \( l \geq 2 \), (43) implies \( y^* < 1 \) when

\[ b < k + l - 1 + \frac{1 - P(l) - lQ(l)}{Q(l)} \left( \frac{P(l)}{(k - 1)Q(l)} \right)^{1/(l-1)}. \] (46)

For parameter values to be in Region 3, both (44) and (46) must be satisfied.

The upper bound on \( b \) from (46) is greater than the lower bound from (44). The upper bound on \( b \) from (46) is less than the upper bound from (44) when

\[ k > \frac{1}{1 - p}. \] (47)

For \( p < 1/2 \), this condition is the same as \( k > 1 \).
Since
\begin{equation}
1 - \frac{P(l) - lQ(l)}{Q(l)} \left( \frac{P(l)}{(k - 1)Q(l)} \right)^{1/(l-1)} > 0,
\end{equation}
any \(k \geq b - l + 1\) always satisfies (46). For \(l = 2\), this rightmost term from (46) reduces to
\begin{equation}
\frac{1}{k - 1},
\end{equation}
which is less than 1 for \(k \geq 2\). Thus, for \(l = 2\), the only solution to (46) is \(k \geq b - l + 1\).

The leftmost term of the right side of (46) \((k)\) increases linearly with \(k\); the rightmost term decreases with \(k\). The rate of decrease slows down as \(k\) increases. As a result, the bound on \(b\) decreases at first and then increases. In some cases the bound also holds for small \(k\), and sometimes it does not; sometimes the small \(k\) region extends all the way to the large \(k\) region, and sometimes it does not.

5.3.4. Region 4. In this case \(x_* < 1\) and \(y_* > 1\). Thus by section 5.3.1, \(k < bP(l - 1)\) and by section 5.3.2, \(k > bP(l) + 1\). Thus, for such \(k\), we have \(bP(l) < k < bP(l - 1)\). In such cases, the candidacy test succeeds with probability near 1, but the frequency test fails with high probability \((F_l \text{ is high})\). Thus, the Apriori Algorithm experiences a bad level in this region. In section 5.4.4 we give bounds on \(F_l\).

5.4. Bounds for \(F_l\). Section 5.3 found the parameters regions relevant to \(F_l\). In this section we will establish bounds on \(F_l\) associated with these regions.

5.4.1. Bounds on \(F_l\) in Region 1. When \(k > bP(l - 1)\) we are in Region 1 of section 5.3. We now give an upper bound on \(F_l\) to show that it is near 0 in this case. Thus in this region the candidacy test fails with high probability.

By (27) with \(y = 1\)
\begin{equation}
F_l \leq x_*^{-k}[1 + (x_*^l - 1)P(l) + l(x_* - 1)Q(l)]^b.
\end{equation}
(Note that bounds on \(F_l\) obtained with \(y = 1\) are also bounds on \(C_l = F_l + S_l\). The definition for \(C_l\) (equation (14)) has a sum over all values of \(j_0\), but setting \(y = 1\) also sums at unit weight over all values of \(j_0\).) The optimum \(x_*\) is given by (35). Solve (35) (with \(y = 1\)) for \(x_*\) with \(x_* = 1 + \delta\) and small \(\delta\). Let \(\theta\) stand for a function that approaches 1 in the limit as \(\delta\) approaches 0. (Just as various big \(O\)'s are associated with different implied constants, different \(\theta\)'s are associated with different functions that approach 1 in the limit.)

\begin{equation}
\delta = \frac{k - bP(l) - bQ(l)}{b[P(l - 1)] - kl[P(l - 1)]} \left( 1 + \frac{[k - bP(l) - bQ(l)][b - k][l(l - 1)P(l)/2]}{b[P(l) + Q(l)] - kl[P(l - 1)]} \right)^{-1}.
\end{equation}

Define \(\alpha_3\) by
\begin{equation}
k = b[P(l - 1) + \alpha_3].
\end{equation}
In (50) replace $k$ by its value in terms of $\alpha_3$ and plug in the value of $x$ implied by (51) to obtain
\begin{equation}
F_l \leq e^{-\beta_0 \alpha_3^2/2\{P(l-1)+(l-1)P(l)-l(P(l-1))^2\}}
\end{equation}
when $\alpha_3$ is small enough, i.e.,
\begin{equation}
\alpha_3 = \{lP(l) + Q(l) - l[P(l-1)]^2\} o(1).
\end{equation}

5.4.2. Region 2. When $k < bP(l) + 1$ we are in Region 2 of section 5.3, and by (34) nearly all item sets pass the frequency test. Since an item set must first pass the candidacy test and then fail the frequency test, $F_l$ can be no larger than $1 - S_l$, which (by (34)) gives the bound
\begin{equation}
F_l \leq e^{-\beta_0 \alpha_2^2/(2P(l)[1-P(l)]) + O(\alpha_2^2 bP(l)^{-2})},
\end{equation}
where $\alpha_2$ is defined by $k = b[P(l) - \alpha_2] + 1$ (equation (33)).

5.4.3. Region 3. Since the $k$ values for Region 3 are entirely inside the union of Regions 1 and 2, we can use results from the previous two sections to obtain upper bounds on $F_l$. With additional algebra, even better upper bounds could be obtained, but the previous bounds are good enough for most purposes.

5.4.4. Region 4. When $bP(l) < k < bP(l-1)$ we are in Region 4 of section 5.3. The candidacy test succeeds with high probability, but the frequency test succeeds with low probability. We now give a lower bound on $F_l$ to show that there are cases where it is near 1.

In (17), the quantity $R_k$ is defined by sums where each $j_i \geq k$ (for $1 \leq i \leq l$). Using inclusion-exclusion arguments, an alternate way to compute $R_k$ is
\begin{equation}
R_k(b, l, m, n) = \sum_h (-1)^h \binom{l}{h} r_k(b, l, m, n, h),
\end{equation}
where
\begin{equation}
r_k(b, l, m, n, h) = \sum_{j_1 < k \atop j_2 < k \atop \vdots \atop j_h < k} \left( \begin{array}{c} b \\ j_1, j_2, \ldots, j_h, b - j_1 - \cdots - j_h \end{array} \right) \\
\times [Q(m)]^{j_1+\cdots+j_h} [1 - P(m) - nQ(m)]^{b-j_1-\cdots-j_h}
\end{equation}
\begin{equation}
= \sum_{j_1 < k \atop j_2 < k \atop \vdots \atop j_h < k} \left( \begin{array}{c} b \\ j_1, j_2, \ldots, j_h, b - j_1 - \cdots - j_h \end{array} \right) \\
\times [Q(m)]^{j_1+\cdots+j_h} [1 - P(m) - (n - l + h)Q(m)]^{b-j_1-\cdots-j_h}.
\end{equation}

The $h = 0$ term of (56) is the sum over the full range for the $j_i$'s. The $h = 1$ term subtracts (for each $j_i$) the part of the range that is not included in the definition of $R$. The $h = 2$ corrects for the overcorrection of the $h = 1$ term (regions where two $j_i$'s were out of range were subtracted off twice). Each successive $h$ corrects for the previous $h$. Therefore, if the sum over $h$ is terminated at some value before $l$, the
result is a lower or upper limit on \( R \) depending on whether the first omitted term is negative or positive. We use the following case of this result:

\[
R_k(b, l, m, n) \geq r_k(b, l, m, n, 0) - lr_k(b, l, m, n, 1)
\]

\[
\geq [1 - P(m) - (n - l)Q(m)]^b - l
\]

\[
\times \sum_{j < k} \binom{b}{j} [Q(m)]^j [1 - P(m) - (n - l + 1)Q(m)]^{b-j}.
\]

By (16) we have

\[
F_l \geq \sum_{j_0 < k} \binom{b}{j_0} [P(l)]^{j_0}
\times \left( [1 - P(l)]^{b-j_0} - l \sum_{j < k-j_0} \binom{b-j_0}{j} [Q(l)]^j [1 - P(l) - Q(l)]^{b-j_0-j} \right).
\]

A lower bound on the sum that comes from the first term in the large parentheses can be obtained by combining (32) with the right part of (11). Applying reasoning similar to that leading to (27) gives the following bound related to the second term:

\[
\sum_{j_0 < k} \binom{b}{j_0} [P(l)]^{j_0} \sum_{j < k-j_0} \binom{b-j_0}{j} [Q(l)]^j [1 - P(l) - Q(l)]^{b-j_0-j}
\]

\[
\leq x^{-k+1} y^{-k+1} [1 + (xy - 1)P(l) + (x - 1)Q(l)]^b,
\]

with the requirement \( x \leq 1, y \leq 1 \). By setting \( y = 1 \) the bound becomes \( x^{-k+1} [1 + (x - 1)P(l - 1)]^b \) (using (10)). The same techniques that give the exponential term in (34) can be applied to bound this term. Combining the bounds on the two parts gives

\[
F_l \geq 1 - e^{-ba_1^2/(2P(l)[1-P(l)]) + O(ba_1^2[1-P(l)]^{-2})} - le^{-ba_4^2/(2P(l-1)[1-P(l-1)]) + O(ba_4^2[1-P(l-1)]^{-2})},
\]

with \( a_1 \) and \( a_4 \) related to \( k \) by \( k = b[P(l) + a_1] \) and \( k = b[P(l - 1) - a_4] - 1 \) when both \( a_1 \) and \( a_4 \) are positive.

This bound is good enough to show that for some values of \( k \), the Apriori Algorithm has one bad level. Consider \( k \) equal to the integer nearest \( [bP(l) + bP(l-1)-1]/2 \), i.e.,

\[
k = \frac{bP(l) + bP(l-1) - 1}{2} + \eta,
\]

with \( |\eta| \leq 1/2 \). This results in

\[
a_1 = \frac{bP(l-1) - P(l) + 1}{2} + \eta, \quad a_4 = \frac{P(l-1) - P(l) + 1}{2} - \eta.
\]

For fixed \( k, p, \) and \( l \) and for large \( b, a_1 \) and \( a_4 \) approach constants. The bound on \( F_l \) (inequality (61)) approaches 1 when \( b \) becomes large. Thus, when \( b \) is large, there is a \( k \) value where \( F_l \) is extremely close to 1. When \( k \) is near \( bP(l) \) for some \( l \) the bound from (63) is not good enough to show that \( F_l \) is close to 1. The sample calculations (in section 7), however, show that for such \( k \) values there are usually two \( l \) values that are each moderately bad (\( F_l \) above a constant), at least when \( b \) is not small. Thus, the conclusion is that the Apriori Algorithm, when it is run on random data, usually has one bad level or two half-bad levels.
6. Total work. Equation (63) shows that for random data there are many cases where the Apriori Algorithm has one bad level, i.e., a level where many item sets pass the candidacy test but few of them pass the frequency test. Equation (41) shows that there are rare cases where the Apriori Algorithm has no bad levels. The Apriori Algorithm has a reputation for being effective in practice [2, 6, 20]. In this section we show that for many parameter values, even when there is a bad level, the bad level comes before the algorithm has done much work and the algorithm is extremely good for the levels after the bad one. This leads to good overall performance. Under the assumption that accesses to the original data dominate the running time, large running times result from those terms in (6) where the binomial coefficient is large and \( S_l + F_l \) is not small. No algorithm that explicitly examines the data to verify the number of occurrences for a set can be fast if a large fraction of the possible sets for large \( l \) must be processed. The merit of the Apriori Algorithm is that \( S_l + F_l \) usually becomes extremely small once \( l \) increases beyond the value that results in \( k > bP(l) \). This is shown by the following rough calculation. Consider the ratio of the \( l \) and \( l + 1 \) terms from (6):

\[
\frac{\binom{|I|}{l+1}}{\binom{|I|}{l}} = \frac{|I| - l}{l + 1} \approx \frac{|I|}{l + 1},
\]

so long as \( l \) is much less than \( |I| \). Now choose \( l \) so that \( k \) is near to \( bP(l-1) \). Using this value of \( l \) in (53) results in \( \alpha_3 \) near 0 and \( F_l \) near 1. Now consider the failure rate on the next level. Using \( l \) to refer to the earlier level, for the level following \( l \), the bound on the failure rate is given by (53) using a value of \( \alpha_3 \) near \( b[P(l-1) - P(l)] \).

\[
F_{l+1} \leq e^{-b^3(l+1)[P(l-1) - P(l)]^2/2(2P(l)+P(l+1)-(l+1)[P(l)]^2)},
\]

Since \( P(l) \) is \( p^l \), for small \( p \) this bound is approximately

\[
e^{-b^3(l+1)p^{-2}/2}.
\]

Since almost every possible item set needs work on level \( l \), the ratio of the amount of work that the Apriori Algorithm performs on level \( l + 1 \) to the amount of work on level \( l \) is approximately

\[
\frac{|I|}{l} e^{-(l+1)b^3/(2p^2)}.
\]

In most interesting cases this ratio will be much less than 1. There is further improvement as \( l \) increases. For most parameter values and for random data the amount of work that the Apriori Algorithm does drops rapidly after the bad level.

7. Sample computations. This section contains sample calculations for \( b = 1024 \) baskets, \( 1 \leq l \leq 5 \), with thresholds in the range \( 1 \leq k \leq 1024 \). Table 1 gives \( S_l \) for \( p = 1/2 \), \( 1 \leq l \leq 5 \). Table 2 gives \( S_l \) for \( p = 1/16 \). Table 3 gives \( F_l \) for \( p = 1/2 \). Table 4 gives \( F_l \) for \( p = 1/16 \). Each table has results for only a few selected values of \( k \). The selected values for \( k \) include those were \( F_l \) is maximum, where it is just above 1/2, and where it is just below 1/2. Figure 1 is a graph of \( S_l \) for \( p = 1/2 \) (solid curves). Upper and lower bounds (dashed curves) from (32) and (34) are also included. Figure 2 is a graph of \( F_l \) (solid curve) for \( p = 1/2 \) along with the bounds from (53), (55), and (63) (dashed). For all bounds plotted in the figures, big \( O \) terms
were ignored and $\theta$ was set to 1. The $p = 1/16$ cases do not lead to clear graphs, so none are given. For this case, one can best see what is happening by examining the tables.

When deciding which results to report, we had to balance the interest in large values for $b$ (up to 100,000 in [2]) with the need to keep the computing time reasonable. Also, we had to balance the interest in small values for $p$ with the need for results to show the various characteristics of the algorithm. And it is difficult to compute $(1 - p)^j$ accurately when $p$ is near zero and $j$ is large. We used code where the number of multiplications increased only as fast as $\ln j$. The values of $S$ were computed exactly using Maple and then converted to floating point. The Maple program was too slow to compute $F$ in this way, so $F$ was computed only with floating point arithmetic. For $S$, the results from exact arithmetic were not significantly different from those for floating point arithmetic, but the floating point calculations sometimes gave zero for values below $10^{-70}$. Also, it was difficult to tell just how close to 1 a floating point value was once it went above 1 - $10^{-12}$.

From Table 1 and also from Figure 1, we see that, for fixed, moderate-sized values of $k$, $S_l$ is extremely close to 1 for small values of $l$ and that $S_l$ is extremely small for large values of $l$. The transition from near 1 to small is quite sharp with increasing $l$. The transition value of $l$ increases as $k$ decreases. For large $k$, even $S_l$ is small. For small $k$ one must go to large $l$ values (not shown) before $S_l$ becomes small. In Figure 1, the red curves (rightmost group) refer to $l = 1$. The upper dotted line is the upper
### Table 2

<table>
<thead>
<tr>
<th>$k$</th>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$S_3$</th>
<th>$S_4$</th>
<th>$S_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$1.0 \times 10^{-0}$</td>
<td>$1.0 \times 10^{-0}$</td>
<td>$1.0 \times 10^{-0}$</td>
<td>$1.0 \times 10^{-0}$</td>
<td>$1.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>2</td>
<td>$2.0 \times 10^{-0}$</td>
<td>$2.0 \times 10^{-0}$</td>
<td>$2.0 \times 10^{-0}$</td>
<td>$2.0 \times 10^{-0}$</td>
<td>$2.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>3</td>
<td>$3.0 \times 10^{-0}$</td>
<td>$3.0 \times 10^{-0}$</td>
<td>$3.0 \times 10^{-0}$</td>
<td>$3.0 \times 10^{-0}$</td>
<td>$3.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>4</td>
<td>$4.0 \times 10^{-0}$</td>
<td>$4.0 \times 10^{-0}$</td>
<td>$4.0 \times 10^{-0}$</td>
<td>$4.0 \times 10^{-0}$</td>
<td>$4.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>5</td>
<td>$5.0 \times 10^{-0}$</td>
<td>$5.0 \times 10^{-0}$</td>
<td>$5.0 \times 10^{-0}$</td>
<td>$5.0 \times 10^{-0}$</td>
<td>$5.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>6</td>
<td>$6.0 \times 10^{-0}$</td>
<td>$6.0 \times 10^{-0}$</td>
<td>$6.0 \times 10^{-0}$</td>
<td>$6.0 \times 10^{-0}$</td>
<td>$6.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>7</td>
<td>$7.0 \times 10^{-0}$</td>
<td>$7.0 \times 10^{-0}$</td>
<td>$7.0 \times 10^{-0}$</td>
<td>$7.0 \times 10^{-0}$</td>
<td>$7.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>8</td>
<td>$8.0 \times 10^{-0}$</td>
<td>$8.0 \times 10^{-0}$</td>
<td>$8.0 \times 10^{-0}$</td>
<td>$8.0 \times 10^{-0}$</td>
<td>$8.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>9</td>
<td>$9.0 \times 10^{-0}$</td>
<td>$9.0 \times 10^{-0}$</td>
<td>$9.0 \times 10^{-0}$</td>
<td>$9.0 \times 10^{-0}$</td>
<td>$9.0 \times 10^{-0}$</td>
</tr>
</tbody>
</table>

### Table 3

<table>
<thead>
<tr>
<th>$F_j$</th>
<th>$F_1$</th>
<th>$F_2$</th>
<th>$F_3$</th>
<th>$F_4$</th>
<th>$F_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$1.0 \times 10^{-0}$</td>
<td>$1.0 \times 10^{-0}$</td>
<td>$1.0 \times 10^{-0}$</td>
<td>$1.0 \times 10^{-0}$</td>
<td>$1.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>2</td>
<td>$2.0 \times 10^{-0}$</td>
<td>$2.0 \times 10^{-0}$</td>
<td>$2.0 \times 10^{-0}$</td>
<td>$2.0 \times 10^{-0}$</td>
<td>$2.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>3</td>
<td>$3.0 \times 10^{-0}$</td>
<td>$3.0 \times 10^{-0}$</td>
<td>$3.0 \times 10^{-0}$</td>
<td>$3.0 \times 10^{-0}$</td>
<td>$3.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>4</td>
<td>$4.0 \times 10^{-0}$</td>
<td>$4.0 \times 10^{-0}$</td>
<td>$4.0 \times 10^{-0}$</td>
<td>$4.0 \times 10^{-0}$</td>
<td>$4.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>5</td>
<td>$5.0 \times 10^{-0}$</td>
<td>$5.0 \times 10^{-0}$</td>
<td>$5.0 \times 10^{-0}$</td>
<td>$5.0 \times 10^{-0}$</td>
<td>$5.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>6</td>
<td>$6.0 \times 10^{-0}$</td>
<td>$6.0 \times 10^{-0}$</td>
<td>$6.0 \times 10^{-0}$</td>
<td>$6.0 \times 10^{-0}$</td>
<td>$6.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>7</td>
<td>$7.0 \times 10^{-0}$</td>
<td>$7.0 \times 10^{-0}$</td>
<td>$7.0 \times 10^{-0}$</td>
<td>$7.0 \times 10^{-0}$</td>
<td>$7.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>8</td>
<td>$8.0 \times 10^{-0}$</td>
<td>$8.0 \times 10^{-0}$</td>
<td>$8.0 \times 10^{-0}$</td>
<td>$8.0 \times 10^{-0}$</td>
<td>$8.0 \times 10^{-0}$</td>
</tr>
<tr>
<td>9</td>
<td>$9.0 \times 10^{-0}$</td>
<td>$9.0 \times 10^{-0}$</td>
<td>$9.0 \times 10^{-0}$</td>
<td>$9.0 \times 10^{-0}$</td>
<td>$9.0 \times 10^{-0}$</td>
</tr>
</tbody>
</table>
bound from (32). The solid one is the actual value from (11). The lower dashed one is the lower bound from (34). Proceeding to the left, we have corresponding groups for \( l = 2, 3, 4, \) and 5. For \( l = 3, 4, \) and 5, one can notice that the plotted “upper bound” goes below the actual value. This is because the big \( O \) term was omitted, and it is significant in these cases. Nonetheless, even without the big \( O \) term the upper bound gives the general idea for how the actual function behaves (it becomes extremely small). Table 2 shows that the \( p = 1/16 \) is similar to the \( p = 1/2 \) case. Notice that \( S \) with \( p = 1/2 \) and \( l = 4 \) has approximately the same value that \( S \) does for \( p = 1/16 \) and \( l = 1 \), particularly when \( k \) is small.

Table 3 and also Figure 2 show the values of \( F_1 \) (solid) for \( p = 1/2 \) from (16), (17), and (19). Figure 3 also shows the upper (dotted) and lower (dashed) bounds computed from (53), (55), and (63). The red group of curves (rightmost) is for \( l = 1 \). The next rightmost group (orange) is for \( l = 2 \); the leftmost group is for \( l = 5 \) (blue). For any fixed \( k \), there is one or sometimes two values of \( l \) for which \( F_1 \) is not small. For most large values of \( k \), there is just one \( l \) value where \( F_1 \) is large, and for that one \( l \) value the resulting \( F_1 \) is extremely close to 1, but for some large \( k \) values, there are two \( l \) values for which \( F_1 \) is moderately large. As \( k \) becomes smaller, the \( l \) value that results in \( F_1 \) being near 1 decreases. Also, \( F_1 \) no longer becomes quite so close to 1.

**Table 4**

<table>
<thead>
<tr>
<th>( k )</th>
<th>( F_1 )</th>
<th>( F_2 )</th>
<th>( F_3 )</th>
<th>( F_4 )</th>
<th>( F_5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( 2.0 \times 10^{-29} )</td>
<td>( 1.8 \times 10^{-2} )</td>
<td>( 7.3 \times 10^{-1} )</td>
<td>( 1.9 \times 10^{-3} )</td>
<td>( 6.4 \times 10^{-10} )</td>
</tr>
<tr>
<td>2</td>
<td>( 1.4 \times 10^{-27} )</td>
<td>( 9.1 \times 10^{-2} )</td>
<td>( 7.3 \times 10^{-1} )</td>
<td>( 2.9 \times 10^{-5} )</td>
<td>( 6.3 \times 10^{-13} )</td>
</tr>
<tr>
<td>3</td>
<td>( 4.8 \times 10^{-26} )</td>
<td>( 2.4 \times 10^{-1} )</td>
<td>( 4.5 \times 10^{-1} )</td>
<td>( 2.3 \times 10^{-7} )</td>
<td>( 3.0 \times 10^{-16} )</td>
</tr>
<tr>
<td>4</td>
<td>( 1.1 \times 10^{-24} )</td>
<td>( 4.3 \times 10^{-1} )</td>
<td>( 2.0 \times 10^{-1} )</td>
<td>( 1.2 \times 10^{-9} )</td>
<td>( 9.8 \times 10^{-20} )</td>
</tr>
<tr>
<td>5</td>
<td>( 1.9 \times 10^{-23} )</td>
<td>( 6.3 \times 10^{-1} )</td>
<td>( 6.3 \times 10^{-2} )</td>
<td>( 4.7 \times 10^{-12} )</td>
<td>( 2.4 \times 10^{-23} )</td>
</tr>
<tr>
<td>22</td>
<td>( 1.5 \times 10^{-10} )</td>
<td>( 1.0 \times 6.0 \times 10^{-10} )</td>
<td>( 7.1 \times 10^{-23} )</td>
<td>( 3.9 \times 10^{-61} )</td>
<td>( 5.6 \times 10^{-93} )</td>
</tr>
<tr>
<td>32</td>
<td>( 2.0 \times 10^{-6} )</td>
<td>( 1.0 \times 4.1 \times 10^{-6} )</td>
<td>( 1.6 \times 10^{-40} )</td>
<td>( 1.7 \times 10^{-93} )</td>
<td>( 2.0 \times 10^{-137} )</td>
</tr>
<tr>
<td>24</td>
<td>( 4.3 \times 10^{-6} )</td>
<td>( 1.0 \times 8.7 \times 10^{-6} )</td>
<td>( 2.1 \times 10^{-42} )</td>
<td>( 8.1 \times 10^{-97} )</td>
<td>( 6.0 \times 10^{-142} )</td>
</tr>
<tr>
<td>45</td>
<td>( 1.7 \times 10^{-3} )</td>
<td>( 1.0 \times 8.4 \times 10^{-3} )</td>
<td>( 2.1 \times 10^{-68} )</td>
<td>( 1.1 \times 10^{-137} )</td>
<td>( 2.7 \times 10^{-197} )</td>
</tr>
<tr>
<td>57</td>
<td>( 1.7 \times 10^{-1} )</td>
<td>( 6.9 \times 10^{-1} )</td>
<td>( 2.4 \times 10^{-92} )</td>
<td>( 5.1 \times 10^{-180} )</td>
<td>( 1.2 \times 10^{-274} )</td>
</tr>
<tr>
<td>58</td>
<td>( 2.0 \times 10^{-1} )</td>
<td>( 6.4 \times 10^{-1} )</td>
<td>( 1.4 \times 10^{-94} )</td>
<td>( 1.3 \times 10^{-183} )</td>
<td>( 1.7 \times 10^{-287} )</td>
</tr>
<tr>
<td>64</td>
<td>( 4.8 \times 10^{-1} )</td>
<td>( 2.7 \times 10^{-1} )</td>
<td>( 3.4 \times 10^{-108} )</td>
<td>( 2.8 \times 10^{-205} )</td>
<td>( 6.4 \times 10^{-209} )</td>
</tr>
<tr>
<td>65</td>
<td>( 5.3 \times 10^{-1} )</td>
<td>( 2.2 \times 10^{-1} )</td>
<td>( 1.7 \times 10^{-110} )</td>
<td>( 6.4 \times 10^{-253} )</td>
<td>( 9.7 \times 10^{-176} )</td>
</tr>
<tr>
<td>91</td>
<td>( 1.5 \times 10^{-4} )</td>
<td>( 3.3 \times 10^{-7} )</td>
<td>( 3.9 \times 10^{-173} )</td>
<td>( 3.9 \times 10^{-253} )</td>
<td>( 4.1 \times 10^{-279} )</td>
</tr>
<tr>
<td>120</td>
<td>( 1.0 \times 5.3 \times 10^{-11} )</td>
<td>( 2.8 \times 10^{-21} )</td>
<td>( 4.0 \times 10^{-250} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
</tr>
<tr>
<td>121</td>
<td>( 1.0 \times 2.6 \times 10^{-11} )</td>
<td>( 6.7 \times 10^{-22} )</td>
<td>( 3.5 \times 10^{-253} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
</tr>
<tr>
<td>128</td>
<td>( 1.0 \times 1.4 \times 10^{-13} )</td>
<td>( 2.1 \times 10^{-26} )</td>
<td>( 9.7 \times 10^{-276} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
</tr>
<tr>
<td>129</td>
<td>( 1.0 \times 6.3 \times 10^{-14} )</td>
<td>( 4.4 \times 10^{-27} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
</tr>
<tr>
<td>186</td>
<td>( 1.0 )</td>
<td>( 7.0 \times 10^{-77} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
</tr>
<tr>
<td>247</td>
<td>( 1.0 )</td>
<td>( 1.3 \times 10^{-100} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
</tr>
<tr>
<td>248</td>
<td>( 1.0 )</td>
<td>( 5.8 \times 10^{-152} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
</tr>
<tr>
<td>256</td>
<td>( 1.0 )</td>
<td>( 5.1 \times 10^{-163} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
</tr>
<tr>
<td>257</td>
<td>( 1.0 )</td>
<td>( 2.0 \times 10^{-164} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
<td>( 4.1 \times 10^{-279} )</td>
</tr>
</tbody>
</table>

8. **Experimental results.** In this section we report the results of running the Apriori Algorithm on data that are more like those used in practice. We used three data sets: (1) synthetic data produced by the generator from the IBM Quest Research...
**Fig. 1.** The value of $S_l$ and approximations to $S_l$ for $p = 1/2$ and $1 \leq l \leq 5$. The red curves are for $S_1$. The upper red dotted curve is the upper bound on $S_1$ (32) with the big $O$ term omitted). The solid red curve is the actual value of $S_1$. The lower red dashed curve is the lower bound on $S_1$ (34) with the big $O$ term omitted). The bounds are plotted only for the range where they are valid. Proceeding to the left, each group of three curves shows similar information on $S_l$ for $l = 2, 3, 4,$ and $5$.

**Fig. 2.** The value of $F_l$ for $p = 1/2$ and $1 \leq l \leq 5$. The leftmost hump is the curve for $l = 5$, the next leftmost hump is for $l = 4$, etc.
Group [13]; (2) real data based on the U.S. Census [23]; and (3) real Web data [24]. These data sets have two major differences from the sets analyzed in the previous sections: (1) the probability of an item being in a basket is not the same for every item, but varies greatly from item to item (this effect is particularly strong in the census and Web data sets), and (2) the items are correlated [5].

Given the major differences between the data used in the analytical study and the data used for the experimental study, it is not surprising that many of the results from the experiments differ from those of the analysis. However, the experiments did verify one important conclusion of the analysis: work done by the Apriori Algorithm is dominated by the failure rate and the total failure rate is almost as high as it possibly can be. To be more precise, for most thresholds, there is a single level where most of
the work is done, and on this level almost every candidate fails to be a success.

In the random model, there is a fixed number of items, $|I|$, each occurring in a
basket with probability $p$. For level $l$, when the threshold, $k$, is such that $k < bp^l$,
almost every item set with $l$ items ($\binom{|I|}{l}$ item sets) is frequent. Once $l$ is large enough
that $k > bp^l$, then almost no item set with $l$ items is frequent. When $p$ is small, the
value of $bp^l$ changes rapidly with $l$.

One can obtain an informal understanding of the experimental data presented
below by permitting $|I|$, $b$, and $p$ to vary (both with the threshold, $k$, and the level, $l$).
In the random model, level $l$ will have approximately $\binom{|I|}{l}$ successful item sets so long
as $k$ is much less than $bp^l$, and it will have a lot less if $k$ is much larger than $bp^l$. Also,
when $p$ is small, the value of $bp^l$ changes rapidly with $l$. The ratio of the number of
item sets on level $l$ to level $l - 1$ is $\binom{|I|}{l}/\binom{|I|}{l-1}$, which is approximately $|I|/l$ if $|I|$ is
much larger than $l$. Thus, we would expect the number of item sets to increase rapidly
with $l$ until the level where $bp^l$ becomes larger than $k$. Even when the effective $|I|
changes with $l$ and the effective $p$ increases, one is likely to get behavior that is
qualitatively the same. The number of item sets will increase rapidly at first, and
then it will suddenly decrease. The cause of the decrease will be the high failure rate
on the level just before the decrease. This will be the level where most of the work is
done, because it is the level with the most candidates.

Now, consider some actual data sets. These data sets were run through a version of
the Apriori Algorithm that computes frequent item sets for all values of the threshold
with a lower limit being imposed on the threshold as needed to avoid running out of
computer memory. Figures 4–9 show the results for three data sets. In each figure, the
bottom axis shows the threshold. Each color curve refers to one level of the Apriori
Algorithm: level 1, red; level 2, orange; level 3, yellow; level 4, green; level 5, blue.
Figures 4, 6, and 8 show, for each of the data sets, the number of candidates as a solid
curve, the upper bound from [10] as a dotted curve, and the number of failures as a
dashed curve. For a given level, the upper bound is highest, the number of candidates
is in the middle, and the number of failures is lowest. Often the dotted and/or dashed
curves are so close to the solid curve that they don’t show up. Figures 5, 7, and 9
show, for each of the data sets, the ratio of failures to candidates (for each level).

8.1. Synthetic data. The first data set was generated using the synthetic data
set generator from the IBM Quest Research Group [13] to create data sets in the same
fashion as [3]. We tested several synthetic data sets, each of which had 1000 items.
We report the results for the T5.I2.D100K (average transaction size, 5; size of the
average maximal frequent item set, 2; number of transactions, 100,000), as they are
representative of the experimental results for the other synthetic data sets.

The solid curves in Figure 4 show the number of candidates on each level (which
is proportional to the work done on the level). Notice that for high thresholds
(above 849) the most work is done on level 1; for intermediate thresholds (between
16 and 849) the most work is done on level 2; and for low thresholds (between 1 and 15)
the most work is done on level 3. Notice that for the highest solid curve (which one
is highest depends on the threshold) the number of failures is almost equal to the
number of candidates. In Figure 4 this shows up by the dashed curve falling on top of
the solid curve (or almost on top). Figure 5 shows the ratio of failures to candidates.
This ratio is close to 1 for the curve that is uppermost in Figure 4.

For each threshold, the uppermost curve represents the bad level, the level where
most of the work is done and where almost every candidate is a failure. For levels past
the bad level, one may want to use a version of the Apriori Algorithm that counts
Fig. 4. Behavior of the algorithm on synthetic data. The dotted lines show the upper bounds, the solid lines show the actual number of candidates considered, and the dashed lines show the number of candidates that fail for each level.

Fig. 5. The failure rate for the synthetic data set.
Fig. 6. Behavior of the algorithm on census data. The dotted lines show the upper bounds, the solid lines show the actual number of candidates considered, and the dashed lines show the number of candidates that fail for each level.

Fig. 7. The failure rate for the census data set.
Fig. 8. Behavior of the algorithm on Web data. The dotted lines show the upper bounds, the solid lines show the actual number of candidates considered, and the dashed lines show the number of candidates that fail for each level.

Fig. 9. The failure rate for the Web data set. Note that each of the first three levels is “bad” for certain ranges of the threshold value.
item sets for all remaining levels in one pass. (This was one of the main motivations for [10].)

8.2. Census data. The second data set was U.S. Census data, using Public Use Microdata Samples (PUMS) [23]. We chose the same sample that was used by [5, 6]. The data represents a 5% sample of the 1990 U.S. Census data for Washington, D.C. It contains 30,370 entries with 122 attributes.

Following [5, 6, 11], we modified the data in the following ways. For monetary values, we took the ceiling of the logarithm of the value. Then we assigned a unique integer to each possible value in the data. In total, the converted PUMS data was 16.6 MB, with 7523 unique items. In addition, we pruned the highly frequent items, which yield an intractable number of frequent item sets.

Due to the high number of correlated items in this type of data our computing resources did not permit us to compute levels 4 and 5 for thresholds less than 25.

The solid curves in Figure 6 show the number of candidates on each level. Notice that for high thresholds (above 872) the most work is done on level 1 and for intermediate thresholds (between 25 and 872) the most work is done on level 2. We are missing data for low thresholds. Figures 6 and 7 show that for the level where most of the work is done, the ratio of failures to candidates is almost 1.

8.3. Web data. The third data set was the BMS-WebView-1 data set described in [24]. This data contains click stream data from a Web-based merchandising company. There are 59,602 transaction and 497 distinct items in the Web data.

Figure 8 shows the results for the Web data. Resources did not permit thresholds below 35 for levels 4 and above.

Level 3, however, has a large enough number of successes to suggest that levels 4 and 5 are the truly “bad” levels for this data. The slope of these two levels is extremely steep, with level 5 being nearly vertical. Figure 9 shows the ratio of failures to candidates for the Web data.

9. Discussion. For random shopper and for most values of the parameters, the amount of work that the Apriori Algorithm does increases rapidly with the level until a bad level is reached. Until the bad level, almost every item set is frequent; level \( l \) will have almost \( \binom{|I|}{l} \) successes. At the bad level and beyond almost every item set fails to be frequent. Almost all of the work done by the algorithm consists of processing the failures on the bad level. Which level is bad depends on the threshold (and the other parameters). For special values of the threshold two neighboring level will each be halfway bad.

The experimental data in the paper came from shoppers that were far from random. For most thresholds, there still was a bad level where almost all the work for the whole algorithm consisted of processing failures on the bad level. The fraction of failures on the early levels was not large, but it was large enough that the number of successes on level \( l \) was much less than \( \binom{|I|}{l} \).

Our assumptions on random shoppers and the assumptions that realize the worst-case bounds of [10, Theorem 1] (highly correlated shopping) appear quite different, but in some ways they are quite similar. Consider the case where the number of successes on level \( l \) is \( \binom{m}{l} \) for some integer \( m \). Then the upper bound limit on the number of candidates for level \( l + 1 \) is \( \binom{m}{l} \), the same as for the random shopper case when \( p = 1 \) and \( |I| = m \). Indeed the random shopper model gives essentially the same prediction for the number of candidates for any value of \( p \) large enough that \( k < bp^l \). From this point of view, the key insight of their Theorem 1 is that the
binomial coefficient leads to a useful estimate of how many items are still important when the Apriori Algorithm gets to level $l$.

We believe that comparing properties that are observed for the random shopper with the properties that are observed with worst-case shoppers can give some insight as to how the Apriori Algorithm will behave with typical data sets.

**Appendix.** This section gives proofs and the derivations of equations.

**Proof.** A subpart of the Apriori Algorithm is $NP$-complete. Determining whether some set of size $l$ occurs $k$ times is in $NP$ because one can guess the set and then verify the number of occurrence by counting the occurrences. The proof that the problem is $NP$-hard uses reduction from the balanced complete bipartite subgraph problem: given a positive integer $K$ and a bipartite graph with vertices $V$ and edges $E$, determine whether there are two disjoint sets of edges ($V_1$ and $V_2$) such that $|V_1| = K$, $|V_2| = K$, and such that there is an edge in $E$ between each vertex in $V_1$ and each vertex in $V_2$. Since any such subgraph must be in a single connected component of the original graph, we can process each connected component of the graph separately. In a single connected component, the vertices of a bipartite graph naturally fall into two groups where all the edges in a group are connected by paths of even length. To map the given single component bipartite graph to baskets and items, associate (in a one-to-one manner) each vertex of one part with an item, and associate (in a one-to-one manner) each vertex of the other part with a basket. Have item $i$ in basket $b$ if and only if the vertex associated with $i$ has an edge connecting to the vertex associated with $b$. If there is solution to the given instance of the balanced complete bipartite subgraph problem, then that solution directly gives an item set of size $K$ that occurs in $K$ baskets. Also if there is an item set of size $K$ that occurs in $K$ baskets, then the corresponding subgraph is a solution to the given instance.

**Equation (11).** We have $j$ ($j \geq k$) baskets that contain the set and $b - j$ baskets that do not contain the set, so

$$S_l = \sum_{j \geq k} \binom{b}{j} [P(l)]^j [1 - P(l)]^{b-j}.$$  

From the binomial theorem we have

$$\sum_j \binom{b}{j} [P(l)]^j [1 - P(l)]^{b-j} = 1,$$

so

$$\sum_{j \geq k} \binom{b}{j} [P(l)]^j [1 - P(l)]^{b-j} = 1 - \sum_{j < k} \binom{b}{j} [P(l)]^j [1 - P(l)]^{b-j}.$$  

**Equation (12).** The factor $[P(l)]^{j_0}$ is the probability that the first $j_0$ baskets obey condition $M_0$, $[Q(l)]^{j_1}$ is the probability that the next $j_1$ baskets obey condition $M_1$, ..., $[Q(l)]^{j_l}$ is the probability that the next $j_l$ baskets obey condition $M_l$, and $[1 - P(l) - lQ(l)]^{b-j_0-j_1-\cdots-j_l}$ is the probability that the remaining baskets obey none of the conditions $M_0, \ldots, M_l$. (Notice that each basket obeys at most one of the conditions $M_h$ ($0 \leq h \leq l$).) However, the various baskets can come in any order, and the multinomial coefficient allows for this. Thus the probability that $j_h$ baskets obey condition $M_h$ ($0 \leq h \leq l$) and that the remaining $b - j_0 - \cdots - j_l$ baskets do not obey
any of the conditions is

\[
(b_{j_0, \ldots, j_l, b - j_0 - \cdots - j_l})[P(l)^{j_0}[Q(l)]^{j_1 + \cdots + j_l}[1 - P(l) - lQ(l)]^{b - j_0 - \cdots - j_l}.
\]

Equation (18).

\[
R_k(b, l, m, n) = \sum_{j_1 \geq k}^{b} \binom{b}{j_1, \ldots, j_l, b - j_1 - \cdots - j_l} \times [Q(m)]^{j_1 + \cdots + j_l}[1 - P(m) - nQ(m)]^{b - j_1 - \cdots - j_l}
\]

(17)

\[
= \sum_{j_1 \geq k}^{b} \binom{b}{j_1} [Q(m)]^{j_1} \sum_{j_2 \geq k}^{b - j_1} \binom{b - j_1}{j_2, \ldots, j_l, b - j_1 - \cdots - j_l} \times [Q(m)]^{j_2 + \cdots + j_l}[1 - P(m) - nQ(m)]^{b - j_1 - j_2 - \cdots - j_l}
\]

(A2)

\[
= \sum_{j \geq k}^{b} [Q(m)]^{j} R_k(b - j, l - 1, m, n).
\]

Equation (19). The boundary condition is just (17) with \( l \) replaced with 0.

Equation (27). Using the binomial theorem on the \( j_1 \) sum in (25), we have

\[
x^{-kt} \sum_{j_1, \ldots, j_l}^{b} \binom{b}{j_1, \ldots, j_l, b - j_1 - \cdots - j_l} \times [xQ(m)]^{j_1 + \cdots + j_l}[1 - P(m) - nQ(m)]^{b - j_1 - \cdots - j_l}
\]

\[
= x^{-kt} \sum_{j_1, \ldots, j_{l-1}}^{x^{-kt}} \binom{b}{j_1, \ldots, j_{l-1}, b - j_1 - \cdots - j_{l-1}} \times [xQ(m)]^{j_1 + \cdots + j_{l-1} + 1}[1 - P(m) - nQ(m)]^{b - j_1 - \cdots - j_{l-1} + 1}.
\]

(A3)

The remaining \( l - 1 \) sums can be done the same way to obtain

\[
R_k(b, l, m, n) \leq x^{-kt} \sum_{j_1, \ldots, j_l}^{b} \binom{b}{j_1, \ldots, j_l, b - j_1 - \cdots - j_l} \times [xQ(m)]^{j_1 + \cdots + j_l}[1 - P(m) - nQ(m)]^{b - j_1 - \cdots - j_l}
\]

\[
\leq x^{-kt}[1 - P(m) - (n - l)Q(m)]^{b}.
\]

Equation (28). Less algebra is needed to minimize the logarithm of the bound, and it leads to the same result. Start with the derivative of the logarithm of (23):

\[
\frac{d}{dx} \{-k \ln x + b \ln[1 + (x - 1)P(l)]\} = \frac{-k}{x} + \frac{bP(l)}{1 + P(l)(x - 1)}.
\]

(A4)

Now set the logarithm to zero and replace \( x \) (the free variable) with \( x_* \) (the value
that results in the derivative being zero):

\[
\frac{-k}{x^*} + \frac{bP(l)}{1 + P(l)(x^* - 1)} = 0,
\]

(A5)

\[-kP(l)(x^* - 1) - k + bP(l)x^* = 0,
\]

(A6)

\[x^* = \frac{k[1 - P(l)]}{(b - k)P(l)}.
\]

Equation (29).

\[
\frac{k[1 - P(l)]}{(b - k)P(l)} > 1,
\]

(A7)

\[k[1 - P(l)] > (b - k)P(l)
\]

(since \(b > k\)), so

(29)

\[k > bP(l).
\]

Equation (30). Plugging (28) into (23) gives

\[
S_l \leq \left(\frac{k[1 - P(l)]}{(b - k)P(l)}\right)^{-k} \left[1 + \left(\frac{k[1 - P(l)]}{(b - k)P(l)} - 1\right)P(l)\right]^b,
\]

(A9)

\[S_l \leq \{k[1 - P(l)]\}^{-k}[P(l)]^k(b - k)^{-b+k}(b - k + \{k[1 - P(l)] - (b - k)P(l)\})^b,
\]

(30)

\[S_l \leq \left(\frac{P(l)}{k}\right) \left(\frac{1 - P(l)}{b - k}\right)^{b-k} b^b.
\]

Equation (32). Replace \(k\) in (30) with its value in terms of \(\alpha_1\) (equation (31)):

\[
S_l \leq \left(\frac{P(l)}{b[P(l) + \alpha_1]}\right)^{b[P(l) + \alpha_1]} \left(\frac{1 - P(l)}{b[1 - P(l) - \alpha_1]}\right)^{b[1 - P(l) - \alpha_1]} b^b,
\]

(A10)

\[S_l \leq \left(\frac{P(l)}{b[P(l) + \alpha_1]}\right)^{b[P(l) + \alpha_1]} \left(\frac{1 - P(l)}{b[1 - P(l) - \alpha_1]}\right)^{b[1 - P(l) - \alpha_1]} b^b,
\]

(A11)

\[S_l \leq \left(\frac{1}{b[1 + \alpha_1/P(l)]}\right)^{b[P(l) + \alpha_1]} \left(\frac{1}{b[1 - \alpha_1/[1 - P(l)]]}\right)^{b[1 - P(l) - \alpha_1]} b^b,
\]

(A12)

\[S_l \leq \left(\frac{1}{1 + \alpha_1/P(l)}\right)^{b[P(l) + \alpha_1]} \left(\frac{1}{1 - \alpha_1/[1 - P(l)]}\right)^{b[1 - P(l) - \alpha_1]} b^b.
\]

(A13)

To further simplify this, we will write it as \(S_l \leq e^X\) with

\[
X = \ln \left(\frac{1}{1 + \alpha_1/P(l)}\right)^{b[P(l) + \alpha_1]} \left(\frac{1}{1 - \alpha_1/[1 - P(l)]}\right)^{b[1 - P(l) - \alpha_1]}
\]

(A14)

\[-b[P(l) + \alpha_1] \ln \left(1 + \frac{\alpha_1}{P(l)}\right) - b[1 - P(l) - \alpha_1] \ln \left(1 - \frac{\alpha_1}{1 - P(l)}\right).
\]

(A15)
Dividing by $b$, we have

\begin{align*}
(A16) \quad \frac{X}{b} &= -[P(l) + \alpha_1] \ln \left(1 + \frac{\alpha_1}{P(l)}\right) - [1 - P(l) - \alpha_1] \ln \left(1 - \frac{\alpha_1}{1 - P(l)}\right) \\
(A17) &= -[P(l) + \alpha_1] \left[\left(\frac{\alpha_1}{P(l)}\right) - \frac{1}{2} \left(\frac{\alpha_1}{P(l)}\right)^2 + O \left(\left(\frac{\alpha_1}{P(l)}\right)^3\right)\right] \\
&\quad + [1 - P(l) - \alpha_1] \left[\left(\frac{\alpha_1}{1 - P(l)}\right) + \frac{1}{2} \left(\frac{\alpha_1}{1 - P(l)}\right)^2\right] \\
&\quad + O \left(\left(\frac{\alpha_1}{1 - P(l)}\right)^3\right) \\
(A18) &= -\alpha_1 + \frac{\alpha_1^2}{2P(l)} - O \left(\alpha_1^3 \frac{1}{P(l)^2}\right) - \frac{\alpha_1^2}{2P(l)^2} - O \left(\alpha_1^3 \frac{1}{P(l)^2}\right) \\
&\quad + \alpha_1 + \frac{\alpha_1^2}{2[1 - P(l)]} + O \left(\alpha_1^3 \frac{1}{[1 - P(l)]^2}\right) - \frac{\alpha_1^2}{1 - P(l)} - \frac{\alpha_1^3}{2[1 - P(l)]^2} \\
(A19) &= -O \left(\frac{\alpha_1^4}{[1 - P(l)]^3}\right) \\
(A20) &= -\frac{\alpha_1^2}{2P(l)} - \frac{\alpha_1^2}{2[1 - P(l)]} + O \left(\frac{\alpha_1^3}{[1 - P(l)]^2}\right) - O \left(\frac{\alpha_1^3}{P(l)^2}\right).
\end{align*}

The big $O$ is with respect to $\alpha_1$. We assume that $0 < p < 1$. Since negative big $O$ terms can be dropped in an upper limit,

\begin{equation}
(32) \quad S_l \leq e^{-\alpha_1^2 / (2P(l)[1 - P(l)]) + O(\alpha_1^3 / (1 - P(l))^2)}.
\end{equation}

Equation (35). The derivative of the logarithm of the bound on $F$ (inequality (27)) with respect to $x$ is

\begin{equation}
(A21) \quad \frac{d[kl \ln x - (k - 1) \ln y + b \ln[1 + (x^l y - 1)P(l) + l(x - 1)Q(l)]]}{dx} = -\frac{k}{x} + \frac{b[x^{l-1} y P(l) + lQ(l)]}{1 + (x^l y - 1)P(l) + l(x - 1)Q(l)}.
\end{equation}

Setting this to zero gives

\begin{align*}
(A22) \quad &-kl[1 + (x_*^l y - 1)P(l) + l(x_* - 1)Q(l)] + bx_*[x_*^{l-1} y P(l) + lQ(l) = 0, \\
(A23) \quad &-kl - klP(l)x_*^l y + klP(l) + kl^2 Q(l) - kl^2 Q(l)x_* + blP(l)x_* y + blQ(l)x_* = 0, \\
(35) \quad & (b - k)P(l)x_*^l y + (b - kl)Q(l)x_* - k[1 - P(l) - lQ(l)] = 0.
\end{align*}

Equation (36). The derivative of the logarithm bound (inequality (27)) with respect to $y$ gives

\begin{equation}
(A24) \quad \frac{d[kl \ln x - (k - 1) \ln y + b \ln[1 + (x^l y - 1)P(l) + l(x - 1)Q(l)]]}{dy} = -\frac{k - 1}{y} + \frac{b x^l P(l)}{1 + (x^l y - 1)P(l) + l(x - 1)Q(l)}.
\end{equation}
Setting this to zero gives

(A26) \[-(k - 1)[1 + (x^l y_0 - 1)P(l) + l(x - 1)Q(l)] + bx^l y_0 P(l) = 0,

(A36) \[(b - k + 1)P(l)x^l y_0 - (k - 1)[1 - P(l) + lQ(l)(x - 1)] = 0.

Equation (37). Equation (35) with \(x_0 = 1, y = 1\) is

(A27) \[(b - k)P(l) + (b - kl)Q(l) - k[1 - P(l) - lQ(l)] = 0,

(A37) \[k = b[P(l) + Q(l)] = bP(l - 1).

Equation (38). By implicit differentiation of (35) (with \(y = 1\)) we have

(A28) \[
\frac{d}{db} \left\{ (b - k)P(l)x_0^l + (b - kl)Q(l)x_0 - k[1 - P(l) - lQ(l)] \right\} = 0,
\]

(A29) \[
P(l)x_0^l + Q(l)x_0 + [(b - k)P(l)x_0^{l-1} + (b - kl)Q(l)] \frac{dx_0}{db} = 0,
\]

(A30) \[
\frac{dx_0}{db} = -\frac{P(l)x_0^l + Q(l)x_0}{(b - k)P(l)x_0^{l-1} + (b - kl)Q(l)}.
\]

Since \(x_0\) solves (35) (with \(y = 1\)) we have

(A31) \[(b - k)P(l)x_0^l + (b - kl)Q(l)x_0 = k[1 - P(l) - lQ(l)],

which is positive. Thus

(A32) \[(b - k)P(l)x_0^{l-1} + (b - kl)Q(l) = \frac{(b - k)P(l)x_0^l + (b - kl)Q(l)x_0}{x_0} \]

\[+ (l - 1)(b - k)P(l)x_0^{l-1},\]

is also positive (because \(x_0 > 0, b - k > 0,\) and \(l \geq 1\)). Thus, \(dx_0/db\) is negative. If we start at the \(b\) value that results in \(x_0 = 1\), and decrease \(b\), then \(x_0\) increases. Thus, it becomes larger than 1 and stays larger than 1.

Equation (39). Equation (36) with \(x = 1\) is

(A33) \[(b - k + 1)P(l)y_0 - (k - 1)[1 - P(l)] = 0,

(A39) \[y_0 = \frac{(k - 1)[1 - P(l)]}{(b - k + 1)P(l)}.
\]

Equation (40).

(A40) \[
\frac{(k - 1)[1 - P(l)]}{(b - k + 1)P(l)} < 1,
\]

(A34) \[(k - 1)[1 - P(l)] < (b - k + 1)P(l),

(A35) \[k - 1 < bP(l),

(A36) \[k < bP(l) + 1.
\]
Equation (42). From (35)

(A37) \[ P(l)x^*_y = \frac{k[1 - P(l) - lQ(l)] - (b - kl)Q(l)x_\ast}{b - k} \]  
From (36)

(A38) \[ P(l)x^*_y = \frac{(k - 1)[1 - P(l) + l(x - 1)Q(l)]}{b - k + 1} \]  
Setting \(x\) to \(x_\ast\), \(y\) to \(y_\ast\), the two right sides equal, and clearing fractions gives

\[ k(b - k + 1)[1 - P(l) - lQ(l)] - (b - kl)(b - k + 1)Q(l)x = (k - 1)(b - k)[1 - P(l) - lQ(l)] \]

(A39) \[ = (k - 1)(b - k)[1 - P(l) + lQ(l)x - lQ(l)], \]

\[ (b^2 - bk + b - bkl + k^2l - kl)Q(l)x = (bk - k^2 + k)[1 - P(l) - lQ(l)] \]

(A40) \[ - (bk - k^2 - b + k)[1 - P(l) + lQ(l)x - lQ(l)], \]

\[ (b^2 - bk + b - bkl + k^2l - kl + bk - k^2l - bl + kl)Q(l)x = (bk - k^2 + k - bk + k^2 + b - k)[1 - P(l) - lQ(l)], \]

(A41) \[ (b^2 - bk + b - bl)Q(l)x = b[1 - P(l) - lQ(l)], \]

(A42) \[ x = \frac{1 - P(l) - lQ(l)}{b - k - l + 1}Q(l) . \]

Equation (43). Plugging the value of \(x\) from (42) into (35) gives

\[ (b - k) \left( \frac{1 - P(l) - lQ(l)}{(b - k - l + 1)Q(l)} \right)^l P(l)y \]

(A43) \[ + \frac{(b - kl)[1 - P(l) - lQ(l)]}{b - k - l + 1} - k[1 - P(l) - lQ(l)] = 0, \]

\[ y = \frac{k[1 - P(l) - lQ(l)] - \frac{(b - kl)[1 - P(l) - lQ(l)]}{b - k - l + 1}}{(b - k) \left( \frac{1 - P(l) - lQ(l)}{(b - k - l + 1)Q(l)} \right)^l P(l) , \]

(A44) \[ y = \frac{(b - k - l + 1)^{l-1}Q(l)^l \{k(b - k - l + 1)[1 - P(l) - lQ(l)] - (b - kl)[1 - P(l) - lQ(l)]\}}{(b - k)[1 - P(l) - lQ(l)]^l P(l)}, \]

(A45) \[ y = \frac{(b - k - l + 1)^{l-1}Q(l)^l(b - k^2 - kl + k - b + kl)[1 - P(l) - lQ(l)]}{(b - k)[1 - P(l) - lQ(l)]^l P(l)}, \]

(A46) \[ y = \frac{(b - k - l + 1)^{l-1}Q(l)^l(k - 1)[1 - P(l) - lQ(l)]}{[1 - P(l) - lQ(l)]^l P(l)}, \]

(A47) \[ y = (k - 1) \left( \frac{(b - k - l + 1)Q(l)}{1 - P(l) - lQ(l)} \right)^l \frac{Q(l)}{P(l)} . \]
Equation (44). To have $x > 1$ we need

\[(A48) \quad \frac{1 - P(l) - lQ(l)}{(b - k - l + 1)Q(l)} > 1.\]

For $b > k + l - 1$ we have

\[(A49) \quad 1 - P(l) - lQ(l) > (b - k - l + 1)Q(l),\]
\[(A50) \quad 1 - P(l) > (b - k + 1)Q(l),\]
\[(A51) \quad b < k - 1 + \frac{1 - P(l)}{Q(l)}.\]

So that the upper and lower bounds look more similar, we rewrite the upper bound on $b$ by adding and subtracting $l$:

\[(44) \quad k + l - 1 < b < k + l - 1 + \frac{1 - P(l) - lQ(l)}{Q(l)}.\]

Suppose $b < k + l - 1$. Then from (A48) we have

\[(A52) \quad 1 - P(l) - lQ(l) < (b - k - l + 1)Q(l),\]
\[(A53) \quad 1 - P(l) < (b - k + 1)Q(l),\]
\[(A54) \quad b > k - 1 + \frac{1 - P(l)}{Q(l)},\]
\[(A55) \quad k + l - 1 > b > k - 1 + \frac{1 - P(l)}{Q(l)}.\]

For this range to be nonempty, we need

\[(A56) \quad k + l - 1 > k - 1 + \frac{1 - P(l)}{Q(l)},\]
\[(A57) \quad 0 > \frac{1 - P(l)}{Q(l)} - l,\]
\[(A58) \quad 0 > \frac{1 - P(l) - lQ(l)}{Q(l)},\]

but this cannot be. We have $P(l) + lQ(l) = p^l + l(1 - p)p^{l-1}$, which are some of the terms in the binomial expansion of $[p + (1 - p)]^l = 1$. Since all of the terms are nonnegative (for $0 \geq p \geq 1$) the sum of some of the terms is no more than 1, so the right side of (A58) is nonnegative. Thus, the range is always empty.

Equation (45).

\[(A59) \quad b < bP(l - 1) + l - 1 + \frac{1 - P(l) - lQ(l)}{Q(l)},\]
\[(A60) \quad b[1 - P(l - 1)] < \frac{1 - P(l) - Q(l)}{Q(l)},\]
\[(A61) \quad b[1 - P(l - 1)] < \frac{1 - P(l - 1)}{Q(l)},\]
\[(A62) \quad b < \frac{1}{Q(l)}.\]
Equation (46). To have \( y < 1 \) we need
\[
(k - 1) \left( \frac{(b - k - l + 1)Q(l)}{1 - P(l) - lQ(l)} \right)^{1/(l-1)} \frac{Q(l)}{P(l)} < 1.
\]
For \( l \geq 2 \)
\[
\frac{(b - k - l + 1)Q(l)}{1 - P(l) - lQ(l)} \left( \frac{(k - 1)Q(l)}{P(l)} \right)^{1/(l-1)} < 1,
\]
\[
b - k - l + 1 \leq \frac{1 - P(l) - lQ(l)}{Q(l)} \left( \frac{(k - 1)Q(l)}{P(l)} \right)^{1/(l-1)},
\]
\[
b < k + l - 1 + \frac{1 - P(l) - lQ(l)}{Q(l)} \left( \frac{P(l)}{(k - 1)Q(l)} \right)^{1/(l-1)}.
\]
The upper bound on \( b \) from (46) is greater than the lower bound from (44).
\[
k + l - 1 + \frac{1 - P(l) - lQ(l)}{Q(l)} \left( \frac{P(l)}{(k - 1)Q(l)} \right)^{1/(l-1)} > k + l - 1,
\]
\[
\frac{1 - P(l) - lQ(l)}{Q(l)} \left( \frac{P(l)}{(k - 1)Q(l)} \right)^{1/(l-1)} > 0.
\]
All the terms on the left are positive for \( 0 < p < 1 \).

Equation (47). We now consider when the upper bound on \( b \) from (46) is less than the upper bound from (44):
\[
k + l - 1 + \frac{1 - P(l) - lQ(l)}{Q(l)} \left( \frac{P(l)}{(k - 1)Q(l)} \right)^{1/(l-1)} < k + l - 1 + \frac{1 - P(l) - lQ(l)}{Q(l)},
\]
\[
\frac{1 - P(l) - lQ(l)}{Q(l)} \left( \frac{P(l)}{(k - 1)Q(l)} \right)^{1/(l-1)} < \frac{1 - P(l) - lQ(l)}{Q(l)},
\]
\[
0 < \frac{1 - P(l) - lQ(l)}{Q(l)} \left[ 1 - \left( \frac{P(l)}{(k - 1)Q(l)} \right)^{1/(l-1)} \right].
\]
As shown above (below (A58)), the first factor is always positive. For the second factor to be positive we need
\[
1 > \left( \frac{P(l)}{(k - 1)Q(l)} \right)^{1/(l-1)},
\]
\[
1 > \frac{P(l)}{(k - 1)Q(l)},
\]
\[
(k - 1)(1 - p)p^{l-1} > p^l,
\]
\[
(k - 1)(1 - p) > p,
\]
\[
k - kp - 1 + p > p,
\]
\[
k(1 - p) > 1,
\]
\[
k > \frac{1}{1 - p}.
\]
Let $x = 1 + \delta$ with small $\delta$ and expand to second order. Let $\theta$ stand for quantities that approach 1 in the limit as $\delta$ approaches 0. In other words, $\theta$ is short for $[1 + o(1)]$, where $\delta$ is the variable that is approaching zero.

\begin{align*}
(\text{A78}) \quad (b - k)P(l)\left(1 + l\delta + \frac{l(l - 1)\delta^2\theta}{2}\right) + (b - kl)Q(l)(1 + \delta) &= k[1 - P(l) - lQ(l)], \\
(\text{A79}) \quad (b - k)P(l)\left(l\delta + \frac{l(l - 1)\delta^2\theta}{2}\right) + (b - kl)Q(l)\delta &= k[1 - P(l) - lQ(l)] - (b - k)P(l) - (b - kl)Q(l), \\
(\text{A80}) \quad \delta &= k[1 - P(l) - lQ(l)] - (b - k)P(l) - (b - kl)Q(l), \\
(\text{A81}) \quad \delta &= \frac{k - bP(l) - bQ(l)}{b[lP(l) + Q(l)] - klP(l) + (b - k)(l - 1)P(l)\theta\delta/2}, \\
(\text{A82}) \quad \delta &= \frac{k - bP(l - 1)}{b[lP(l) + Q(l)] - klP(l - 1) + (b - k)(l - 1)P(l)\theta\delta/2}, \\
(\text{A83}) \quad \delta &= \frac{k - bP(l - 1)}{b[lP(l) + Q(l)] - klP(l - 1)} \left(1 + \frac{(b - k)(l - 1)P(l)\theta\delta/2}{b[lP(l) + Q(l)] - klP(l - 1)}\right)^{-1}, \\
(51) \quad \delta &= \frac{k - bP(l - 1)}{b[lP(l) + Q(l)] - klP(l - 1)} \left(1 + \frac{|k - bP(l - 1)|(b - k)(l - 1)P(l)\theta/2}{b[lP(l) + Q(l)] - klP(l - 1)^2}\right)^{-1}.
\end{align*}

Equation (53). Write (50) as

\begin{align*}
(\text{A84}) \quad F_l &\leq e^X,
\end{align*}

with

\begin{align*}
(\text{A85}) \quad X &= \ln\{x^{-kl}[1 + (x^l - 1)P(l) + l(x - 1)Q(l)]^b\} \\
(\text{A86}) \quad \quad &= -kl \ln x + b \ln\{1 + (x^l - 1)P(l) + l(x - 1)Q(l)\}.
\end{align*}

Replace $x$ with $1 + \delta$:

\begin{align*}
(\text{A87}) \quad X &= -kl \ln(1 + \delta) + b \ln\{1 + [(1 + \delta)^l - 1]P(l) + lQ(l)\delta\}.
\end{align*}
Expanding $X$ in a power series to second order gives

$$X = -kl \ln(1 + \delta) + b \ln \left(1 + l\delta P(l) + \frac{l(l-1)P(l)\delta^2}{2} + lQ(l)\delta\right)$$

$$= -kl\delta + \frac{k\theta\delta^2}{2} + b \left(lP(l)\delta + \frac{l(l-1)\delta^2P(l)\theta}{2} + lQ(l)\delta\right)$$

(A88) \hspace{1cm} - \frac{b}{2} \left(lP(l)\delta + \frac{l(l-1)P(l)\theta\delta^2}{2} + lQ(l)\delta\right)^2 \theta$$

(A89) \hspace{1cm} = -l[k - bP(l) - bQ(l)]\delta + \frac{kl + bl(l-1)P(l) - bl^2[P(l) + Q(l)]}{2} \delta^2$$

(A90) \hspace{1cm} = -l[k - bP(l - 1)]\delta + \frac{kl + bl(l-1)P(l) - bl^2[P(l - 1)]}{2} \delta^2.$$

Replace $k$ by its definition in terms of $\alpha_3$ (equation (52)) to obtain

(A91) \hspace{1cm} X = -l[k - bP(l - 1)]\delta + \frac{kl + bl(l-1)P(l) - bl^2[P(l - 1)]^2}{2} \delta^2$

(A92) \hspace{1cm} = -b\alpha_3\delta + \frac{bl\{P(l - 1) + \alpha_3 + (l-1)P(l) - l[P(l - 1)]\}}{2} \delta^2$

(A93) \hspace{1cm} = \left(-b\alpha_3 + \frac{bl\{P(l - 1) + \alpha_3 + (l-1)P(l) - l[P(l - 1)]\}}{2} \delta\right) \delta.$$

Also replace $k$ in (51) by its value in terms of $\alpha_3$ to obtain

$$\delta = \frac{\alpha_3 b}{b[P(l) + Q(l)] - b[P(l - 1) + \alpha_3]P(l - 1)}$$

(A94) \hspace{1cm} \times \left(1 + \frac{b\alpha_3\{b - b[\alpha_3\{P(l - 1) + \alpha_3\}]l\theta}{b[P(l) + Q(l)] - b[P(l - 1) + \alpha_3]l[P(l - 1)]}\right)^{-1}$

$$= \frac{\alpha_3}{lP(l) + Q(l) - l[P(l - 1) + \alpha_3]P(l - 1)}$$

(A95) \hspace{1cm} \times \left(1 + \frac{\alpha_3[1 - P(l - 1) - \alpha_3]l\theta}{lP(l) + Q(l) - l[P(l - 1) + \alpha_3]l[P(l - 1)]}\right)^{-1}.$$

Since $\delta$ and $\alpha_3$ go to zero together, this can be written as

(A96) \hspace{1cm} \delta = \frac{\theta\alpha_3}{lP(l) + Q(l) - l[P(l - 1)]^2}. 

\[\]
r is equal to the sum over the entire area (equation (A99)) gives

(A97) \[
X = \left(-b\alpha_3 + \frac{bl\{P(l-1) + \alpha_3 + (l-1)P(l) - l[P(l-1)]^2\} \theta \alpha_3}{2}\right) \delta
\]

(A98) \[
\times \frac{P(l-1) + (l-1)P(l) - l[P(l-1)]^2}{\theta \alpha_3}
\]

(A99) \[
= \left(-b\alpha_3 + \frac{bl\{P(l-1) + (l-1)P(l) - l[P(l-1)]^2 + \alpha_3\} \theta \alpha_3}{2\{P(l-1) + (l-1)P(l) - l[P(l-1)]^2\}}\right)
\]

(A100) \[
\times \frac{P(l-1) + (l-1)P(l) - l[P(l-1)]^2}{bl\theta \alpha_3^2}
\]

Thus,

(Equation 53) \[
F_l \leq e^{-b\alpha_3^2/(2\{P(l-1) + (l-1)P(l) - l[P(l-1)]^2\})}.
\]

Equation (54). The derivation of (53) requires that \(\delta = o(1)\). The step from (A98) to (A99) requires that \(\alpha_3\) be small compared to some other terms. Both conditions imply

(Equation 54) \[
\alpha_3 = \{lP(l) + Q(l) - l[P(l-1)]^2\} o(1).
\]

Equation (56). By inclusion-exclusion, the sum for the region that defines \(R_k\) is equal to the sum over the entire area \((r_k(b,l,m,n,0))\), minus the sums over the various regions where a single \(j\) is required to be outside of \(R_k\)'s region \((l\) copies of \(r_k(b,l,m,n,1))\), plus the sums over regions where two \(j\)'s are required to be outside of \(R_k\)'s region, etc.

Equation (58).

\[
r_k(b,l,m,n,h) = \sum_{j_1 < k, j_2 < k, j_3 < k, j_h < k, j_{h+1},...,j_l} \binom{b}{j_1, j_2, ..., j_i, b - j_1 - \cdots - j_i}
\]

\[
\times [Q(m)]^{j_1 + \cdots + j_i} [1 - P(m) - nQ(m)]^{b - j_1 - \cdots - j_i}
\]

\[
= \sum_{j_1 < k, j_2 < k, j_3 < k, j_h < k, j_{h+1},...,j_{l-1}} \binom{b}{j_1, j_2, ..., j_{l-1}, b - j_1 - \cdots - j_{l-1}}
\]

\[
\times [Q(m)]^{j_1 + \cdots + j_{l-1}} [1 - P(m) - (n - 1)Q(m)]^{b - j_1 - \cdots - j_{l-1}}
\]

\[
\cdots
\]

\[
= \sum_{j_1 < k, j_2 < k, j_3 < k, j_h < k} \binom{b}{j_1, j_2, ..., j_h, b - j_1 - \cdots - j_h}
\]

\[
\times [Q(m)]^{j_1 + \cdots + j_h} [1 - P(m) - (n - l + h)Q(m)]^{b - j_1 - \cdots - j_h}.
\]
Bound on first term of (61). For
\[ \sum_{j_0 < k} \binom{b}{j_0} [P(l)]^{j_0} [1 - P(l)]^{b-j_0} \]
use the Chernoff bound from (23).

Equation (62).

\[
\sum_{j_0 < k} \binom{b}{j_0} [P(l)]^{j_0} \sum_{j < k-j_0} \binom{b-j_0}{j} [Q(l)]^j [1 - P(l) - Q(l)]^{b-j_0-j} \\
\leq x^{-k+1} y^{-k+1} \sum_{j_0, j} \binom{b}{j_0} [xyP(l)]^{j_0} \binom{b-j_0}{j} \\
xQ(l)]^j [1 - P(l) - Q(l)]^{b-j_0-j} \\
\leq x^{-k+1} y^{-k+1}[1 + (xy - 1)P(l) + (x - 1)Q(l)]^b.
\]

In real life, the Apriori Algorithm is used to analyze data that are more complex. Presumably, no one is interested in running the algorithm on truly random data. Rather, one is interested in the way in which the data differ from random. Nonetheless, we believe that analysis with this simple probability model brings out the main features of the performance of the algorithm. In principle, the techniques in this paper can be applied to more complex probability models of shopping. The challenge is to carry out the resulting calculations so that one can understand the implications of the formulas that result when the analysis is done on more general probability models.

With our probability model we calculate two quantities:
1. **Success rate**: the probability that a set is a success (i.e., passes the frequency test) and test.
2. **Failure rate**: the probability that a set is a candidate (i.e., passes the candidacy test) but not a success (i.e., fails the frequency test).

Notice that the success rate is a property of the probability model, not the algorithm. All correct algorithms will have the same success rate. The Apriori Algorithm never believes that an item set occurs \( k \) times without verifying the fact by counting occurrences in the data base. For algorithms that use this approach, the success rate represents unavoidable work. The Apriori Algorithm is clever in trying to reduce the failure rate. The failure rate represents work that one might hope to avoid.

Also, when there is a total of \( b \) baskets, \( b_A \) baskets with item \( A \), and \( b_B \) items with item \( B \), the Apriori Algorithm is not aware that there must be at least \( 2b - b_A - b_B \) baskets that contain both items \( A \) and \( B \) \([11]\). Similar ideas are explored in \([7]\).

In the best case, the test for candidacy correctly predicts the result of the test for frequency, and the amount of time spent by the Apriori Algorithm is essentially the amount of time spent verifying that all the output sets should be output. A naive upper bound comes from the fact that each candidate is based on a previous success. Each success can lead to at most \( |I| \) candidates. Many previous papers have focused on the total amount of work done by the Apriori Algorithm without concern as to the amount of output generated, but such studies can be misleading in that the amount of output is the main factor that determines how much work an Apriori-like algorithm will do, and the amount of output is a feature of the problem instance rather than of the algorithm. The simple lower and upper bound analyses say that the ratio between
the number of candidates and the number of successes (i.e., the output) is between 1 and \(|I|\) (so long as there is at least one success).

Stronger worst-case bounds are given by Geerts, Goethals, and Van den Bussche [10]. They start by writing the number of successes on level \(l\) as the sum of distinct binomial coefficients where the first binomial coefficient has \(l\) as its bottom index and the largest possible top index. The remaining terms (if any) have smaller bottom indices. Their Theorem 1 says that the number of candidates on level \(l+1\) is bounded by the number obtained from increasing each bottom index by 1. This bound is exact for some distributions, including one that results in no failures. Experiments show that in many cases this upper bound is close to the exact value.

At first sight, the approach of Theorem 1 of [10] looks quite different from the approach in this paper. However, in those cases where the number of candidates on level \(l\) is exactly \(\binom{m_{l,t}}{l}\) for some \(m_{l,t}\) there are just two differences. First, [10] considers only items that might still be active during the current level of the Apriori Algorithm \(m_{l,t}\). This is their main insight. Second, the upper limit corresponds to setting our \(p\) to 1. The work in [10] shows that the number of candidates on level \(l\) can never be more than \(\binom{m_{l,t}}{l}\). If \(m_{l,t}\) is much bigger than \(l\), the ratio of candidates on level \(l+1\) to candidates on level \(l\) is approximately \(m_{l,t}/(l+1)\). When the number of candidates cannot be represented as an appropriate binomial, there are additional technical differences between the approach in [10] and in this paper, but they are not significant to a qualitative understanding of the situation.

It is not logically necessary that an algorithm verify occurrences by explicit counting. One alternative algorithm uses ideas that are the complement of those used by the Apriori Algorithm. The key idea for this complementary algorithm is that if some superset of a set \(J\) occurs at least \(k\) times, then so does set \(J\). Also, if one changes the problem so that one needs only the maximal frequent item sets instead of all frequent item sets, then the amount of output needed is greatly reduced for some problem instances.

**Acknowledgments.** The authors want the express their thanks to Nele Dexters for helpful discussions and careful proofreading. They would also like to acknowledge the support of Indiana University Research and Technology Services for computing support during the experiments.

**REFERENCES**


