$$> ml - m \sum_{i=0}^{\infty} \left(1 - \frac{1}{m}\right)^{2^i}$$

$$> ml - m \lceil lg\ m \rceil.$$

It follows that

$$\mathbf{E}[\mathbf{r}] > m(\mathbf{E}[\mathbf{lpsl}] - \lceil lg\ m \rceil).$$

Since V_{l-1} is just some subset of R then $\mathbf{E}[\mathbf{N}] \geq \mathbf{E}[\mathbf{r}]$ and

$$m\mathbf{E}[\mathbf{psl}] > m(\mathbf{E}[\mathbf{lpsl}] - \lceil lg\ m \rceil).$$

Solving for $\mathbf{E}[\mathbf{lpsl}]$ we get

$$\mathbf{E}[\mathbf{lpsl}] < \mathbf{E}[\mathbf{psl}] + \lceil lg\ m \rceil.$$

The lower bound holds because $\mathbf{psl} \leq \mathbf{lpsl}$ for every table . □

Corollary: *The expected value of* **lpsl** *for a full Robin Hood hash table is* $\Theta(\ln n)$.

Proof: Follows from the theorem and the fact that $\mathbf{E}[\mathbf{psl}] = \Theta(\ln n)$ for a full Robin Hood hash table. □

Our extensive experiments indicate that, the longest sequence for full tables is about $1.15\ln n + 2.5$. The bounds, and so the heuristic, are within a constant factor of what we would get if the table is organized to minimize the length of the longest probe sequence. These lower bounds are $\lceil -\alpha^{-1}\ln(1-\alpha) \rceil$ and $\ln n + \gamma + o(1)$ but require solving the appropriate assignment problem in potentially $\Theta(n^2 \log n)$ time [G].

Based on the initial thesis, that a major drawback to the usual insertion scheme is the high variance, we feel we have made an interesting improvement. However if the probe choice of a record cannot be computed in constant time from a key value and the location in which it is found, the cost of completely filling a table will increase from $\Theta(n \log n)$ for the standard algorithm to $\Theta(n \log^2 n)$ for the Robin Hood heuristic. (This is based on determining the probe choice by performing a standard search for the record). The other difficulty is, of course, that no improvement has been made in the mean search time. Both problems are addressed in the next section.

4. Smart Searching

Given that we know the expected position of an element with respect to its probe sequence (either from the theorems or by explicitly keeping track of this value) it seems reasonable to look first in that expected position. More formally, we propose the following search procedure:

> Let t denote ($\sum$ length probe sequence /# elements); *pfirst*, the length of the shortest probe sequence (this will probably be 1); and *plast*, the length of the longest. Then to search for a key first try its t-th choice (this is trivially computable for methods such as double hashing), then $t+1$, $t-1$, $t+2$, $t-2$, . . . to cover the range *pfirst* to *plast*. If the element is not found within this range, it is not present in the table.

For lack of a better name we will call this approach smart search. The expected number of probes for an unsuccessful search is, of course, no more than under the conventional scheme. On the other hand, successful searches are substantially improved as a consequence of the extremely low variance.

Theorem 4. *The expected number of probes for a successful search using the smart search algorithm above is*

$$\sum_{i=1}^{t} (2(t-i)+1)(p_{i+1}-p)$$
$$+ \sum_{i=t+1}^{2t} 2(i-t)(p_{i+1}-p_i) + \sum_{i=2t+1}^{n} (i-t)(p_{i+1}-p_i)$$

where $t = \lfloor \ln(n) + \gamma \rfloor$ *for a full table, and* $t = \lfloor -\alpha^{-1}\ln(1-\alpha) \rfloor$ *for a table with load factor* α.

There are several possible variations of the above search heuristic. The initial probe choice, t, can be estimated slightly differently and some sequence other than $t+1$, $t-1$, $t+2$, $t-2$, . . . can be used.

Define a mean centered search heuristic to be one that finds a key that is k positions above (or below) the mean within ck probes, for some constant c. We then have the following:

Theorem 5: *Any mean centered approach for searching a Robin Hood hash table has an expected search cost of* $O(1)$.