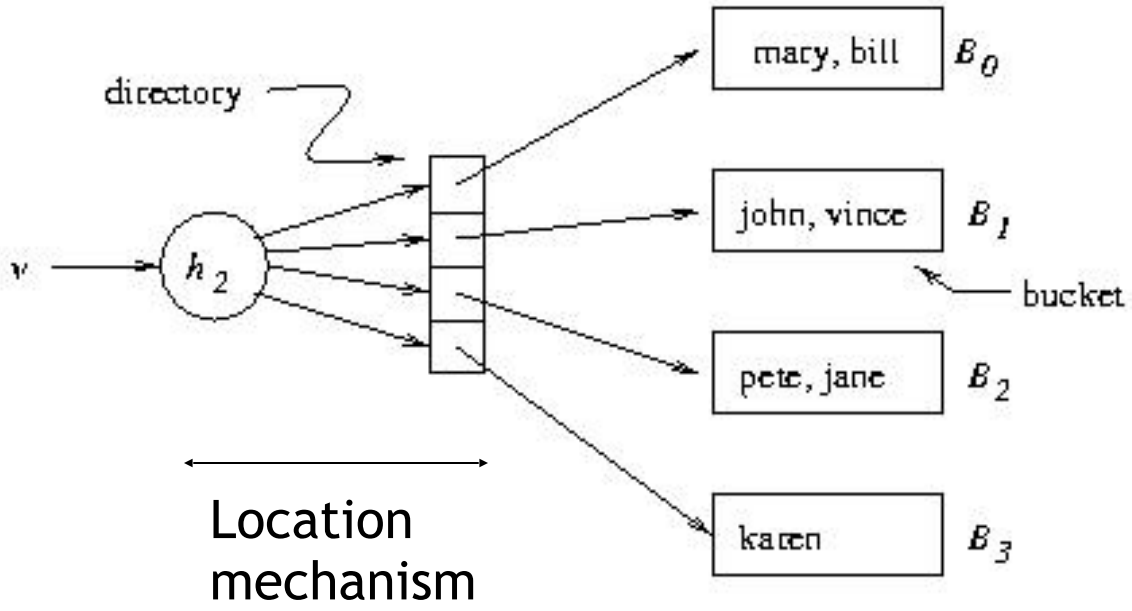
Indexes

- Additional data structure used to reduce the pages accesses necessary to find a row or rows
- Search Key
- Search Key is not necessarily unique
- Location Mechanism
  - Algorithm+Data Structure

Extendable Hashing

- Type of hashing that eliminates chains of pages caused by collisions
- Range of hash function has to be extended to accommodate additional buckets
- **Example:** family of hash functions based on h:
  - $h_k(v) = h(v) \mod 2^k$ (use the last k bits of $h(v)$)
  - At any given time a unique hash, h_k , is used depending on the number of times buckets have been split

Extendable Hashing – Example



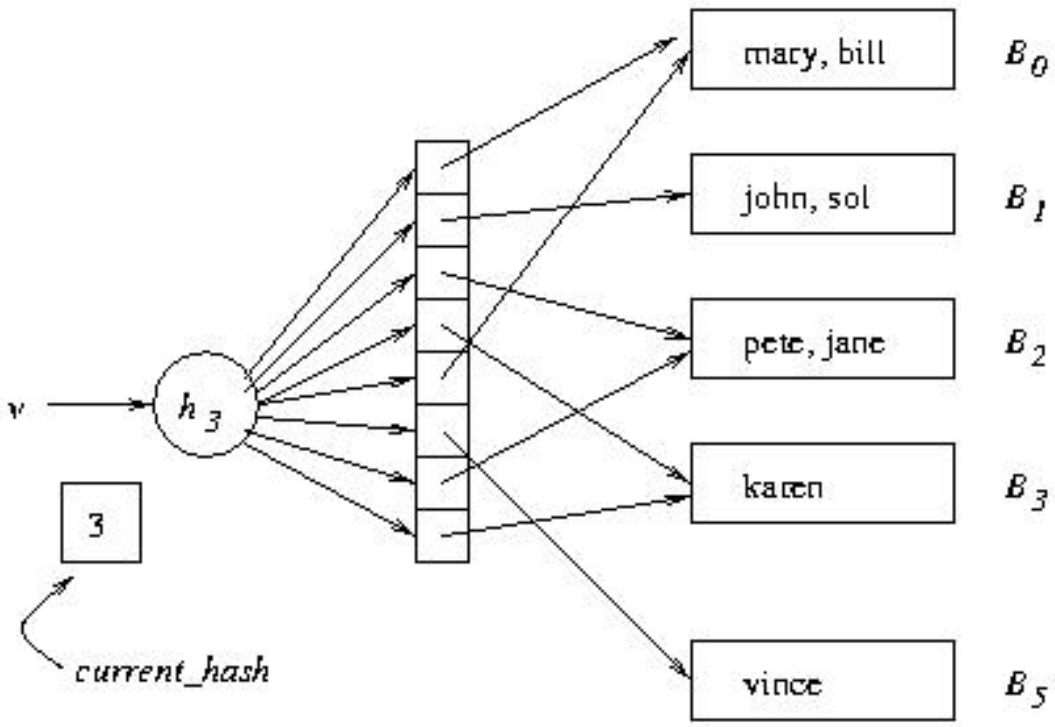
v	*h*(*v*)
pete	11010
mary	00000
jane	11110
bill	00000
john	01001
vince	10101
karen	10111

Extendable hashing uses a directory (level of indirection) to accommodate family of hash functions

Suppose next action is to insert sol, where *h*(*sol*) = *10001*.

Problem: This causes overflow in B_1

Example (cont'd)

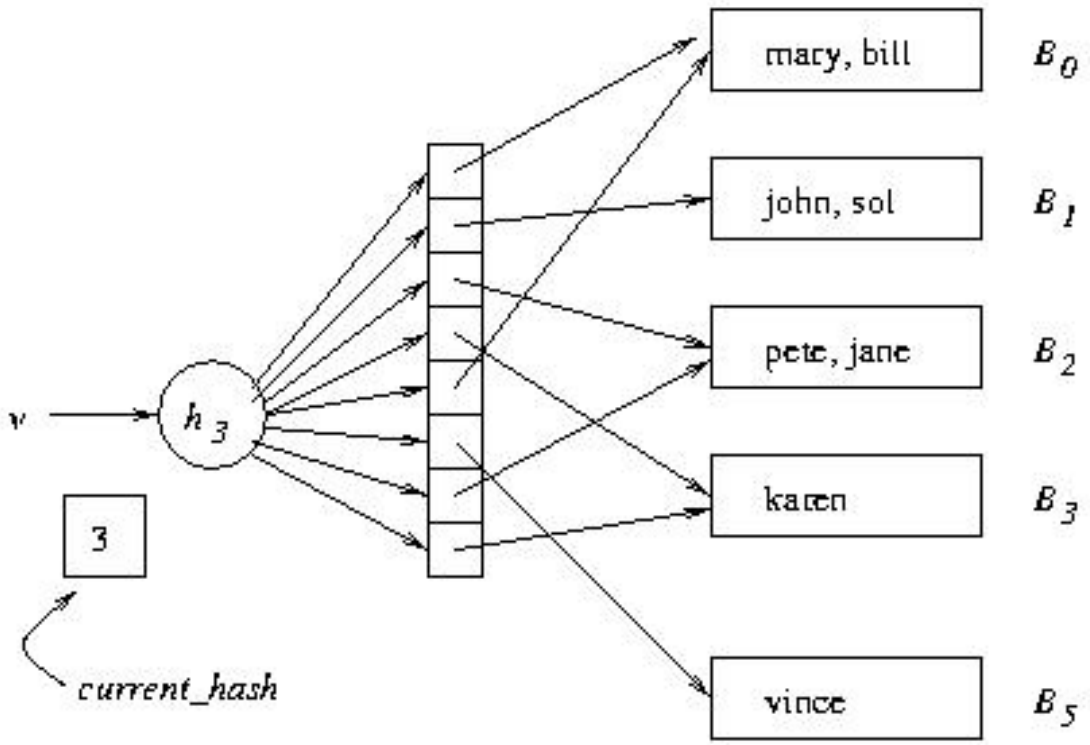


Solution:

1. Switch to h_3
2. Concatenate copy of old directory to new directory
3. Split overflowed bucket, *B*, into *B* and *B′*, dividing entries in *B* between the two using h_3
4. Pointer to *B* in directory copy replaced by pointer to *B′*

Note: Except for *B′* , pointers in directory copy refer to original buckets.
current_hash identifies current hash function.

Example (cont'd)

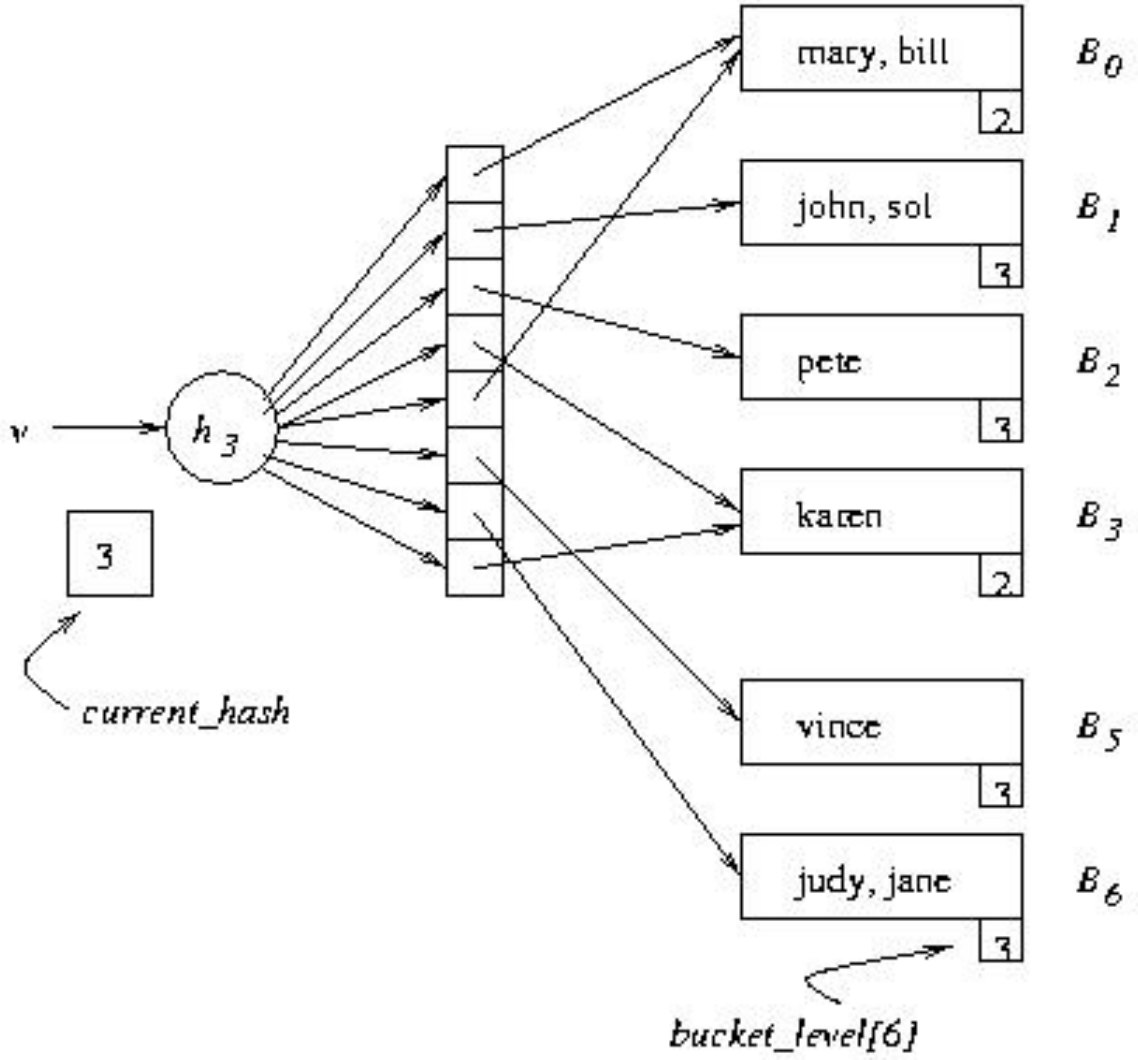


Next action: Insert judy, where $h(judy) = 00110$
B_2 overflows, but directory need not be extended

Problem: When B_i overflows, we need a mechanism for deciding whether the directory has to be doubled

Solution: *bucket_level[i]* records the number of times B_i has been split. If *current_hash* > *bucket_level[i]*, do not enlarge directory

Example (cont'd)



Extendible Hashing Problem

What does an extendable hash table with a bucket size of 2 look like after the following values are inserted? Assume the starting table has 2 buckets and used h_1

Key	Hash Value
10	100010
323	101001
90	111011
80	001101
37	110111
205	010100
100	000110
120	110110