UPPSALA
UNIVERSITET

Advanced Programming

Hash tables

Sara Stymne

Based on slides by Markus Saers

UPPSALA
UNIVERSITET

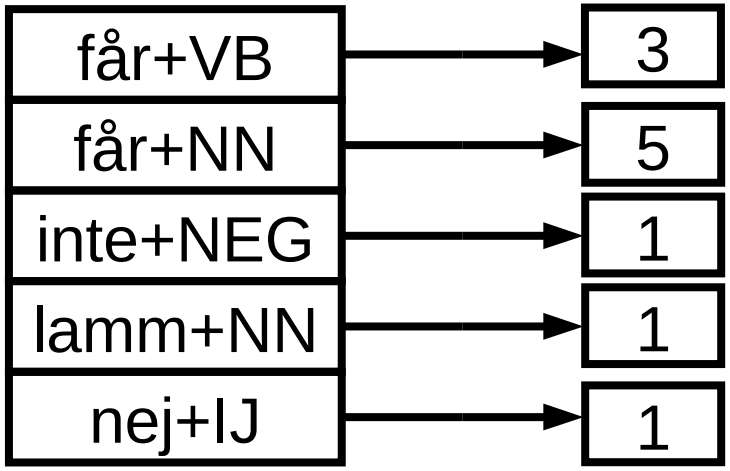
Maps (associative data structures)

- Two common implementations
  - Hash tables
    - (Very) fast
      - O(1) – amortized
    - Unsorted
  - Search trees
    - Fast
      - O(log n) (if balanced)
    - Sorted

UPPSALA
UNIVERSITET

Hash tables

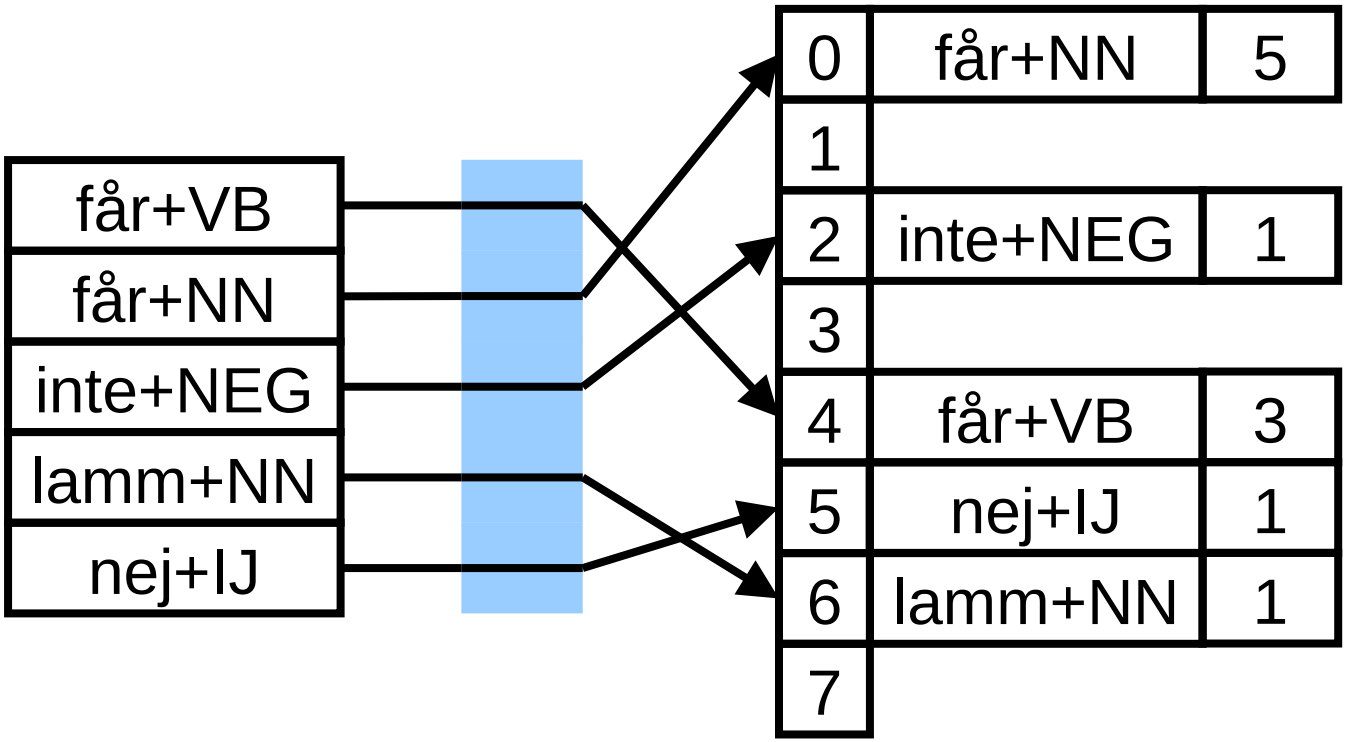
- key–value pair



UPPSALA UNIVERSITET

Hash tables

- Hash function
  - Converts keys to valid indeces
  - Internally the values are stored in an array



UPPSALA
UNIVERSITET

Hash tables

- Insertion
  - Use the hash function
  - Insert the value at the position of the index
- Search
  - Use the hash function
  - Lookup this index

UPPSALA
UNIVERSITET

Hash function and array size

- The hash function returnes an arbitrary integer, x
- The array has a given size, n
- How can we assure that the hash value is restricted to the array size?
  - 0 <= x < n

UPPSALA
UNIVERSITET

Hash function and array size

- The hash function returns an arbitrary integer, x
- The array has a given size, n
- How can we assure that the hash value is restricted to the array size?
  - 0 <= x < n
- Modulo!
- hashCode % n
  - Guarantees that the hash values is in the interval 0 <= x < n

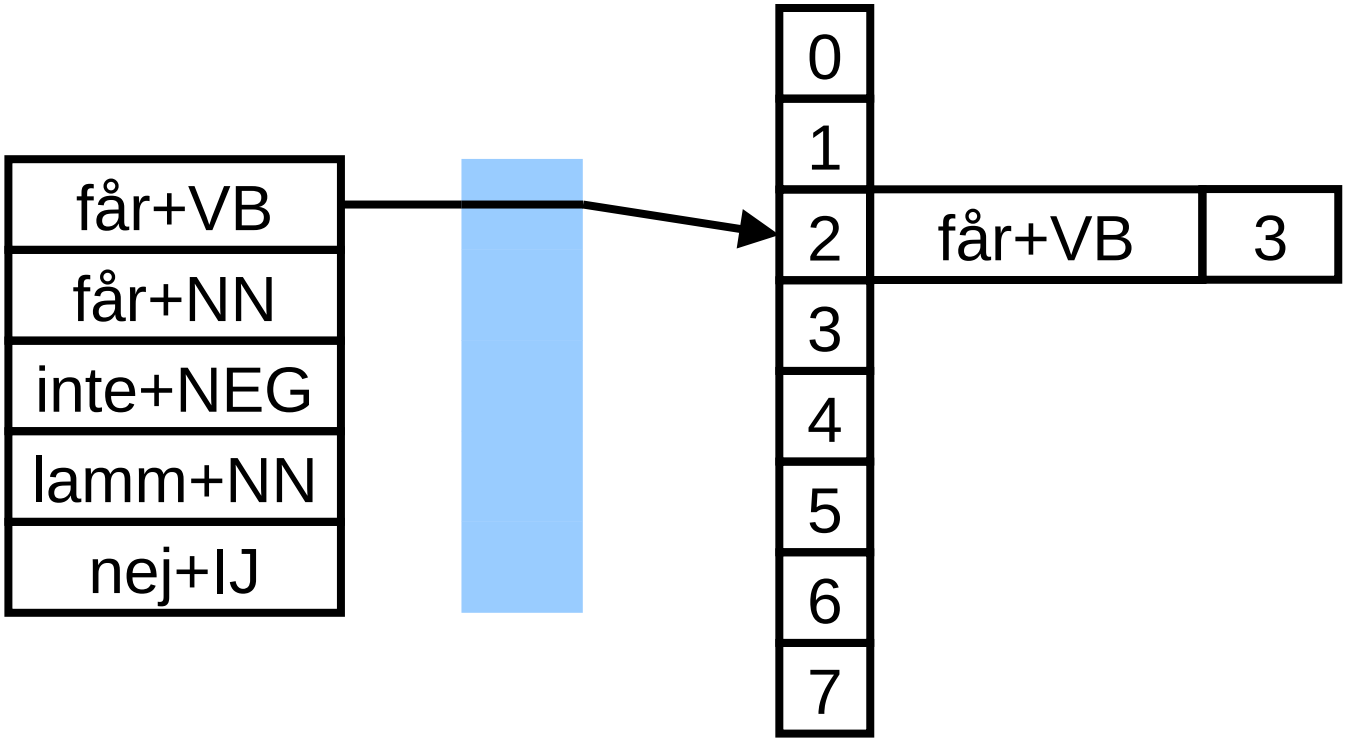
UPPSALA
UNIVERSITET

Collision handling

- Insertion
  - Use the hash function
  - Insert the value at the position of the index
- Search
  - Use the hash function
  - Lookup this index
- **Collision handling**
  - What happens if two keys get the same hash value?

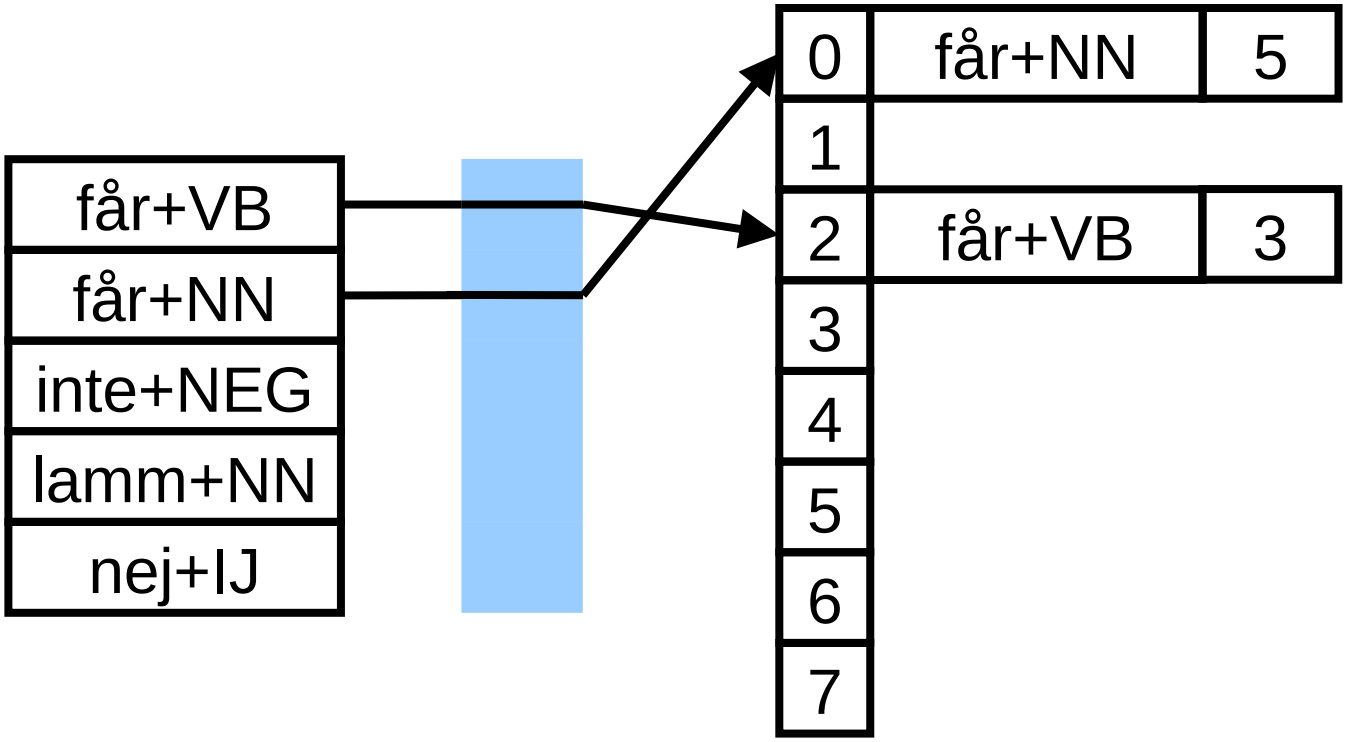
UPPSALA UNIVERSITET

Hash tables: collisions



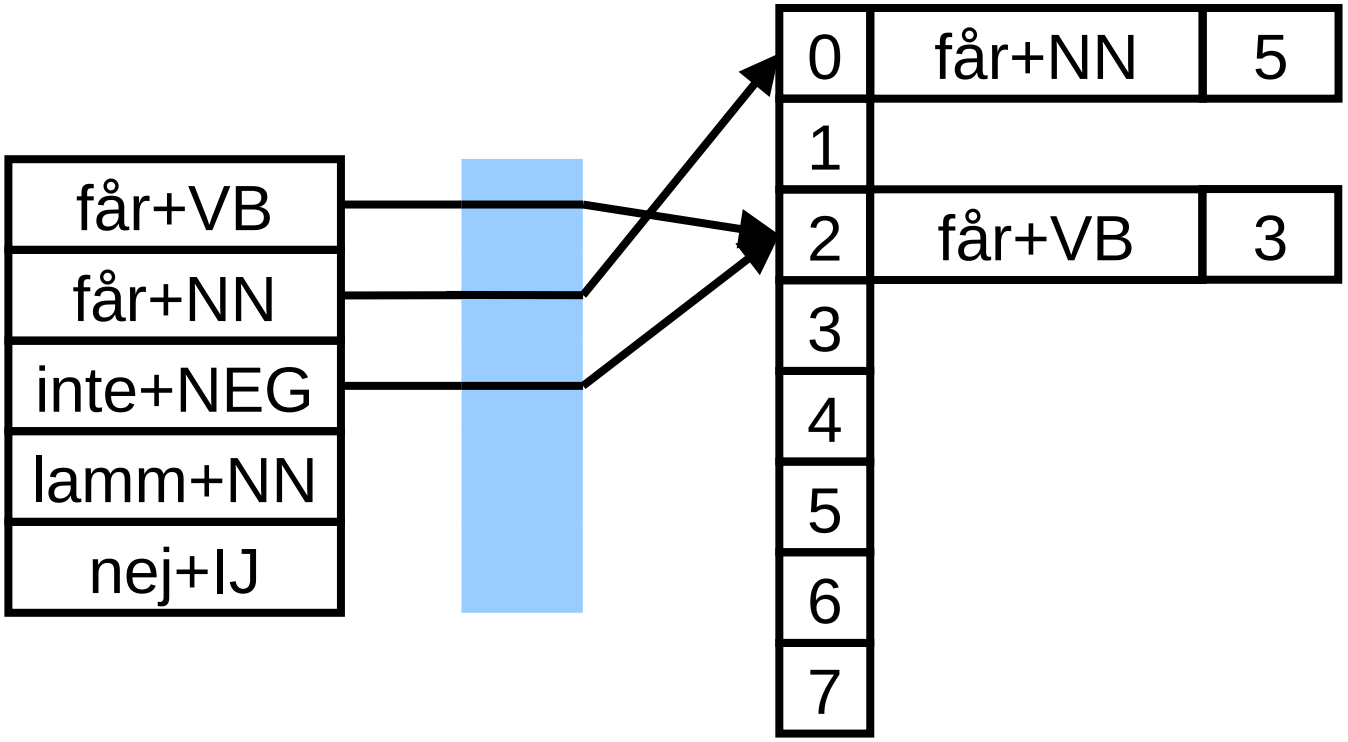
UPPSALA UNIVERSITET

Hash tables: collisions



UPPSALA UNIVERSITET

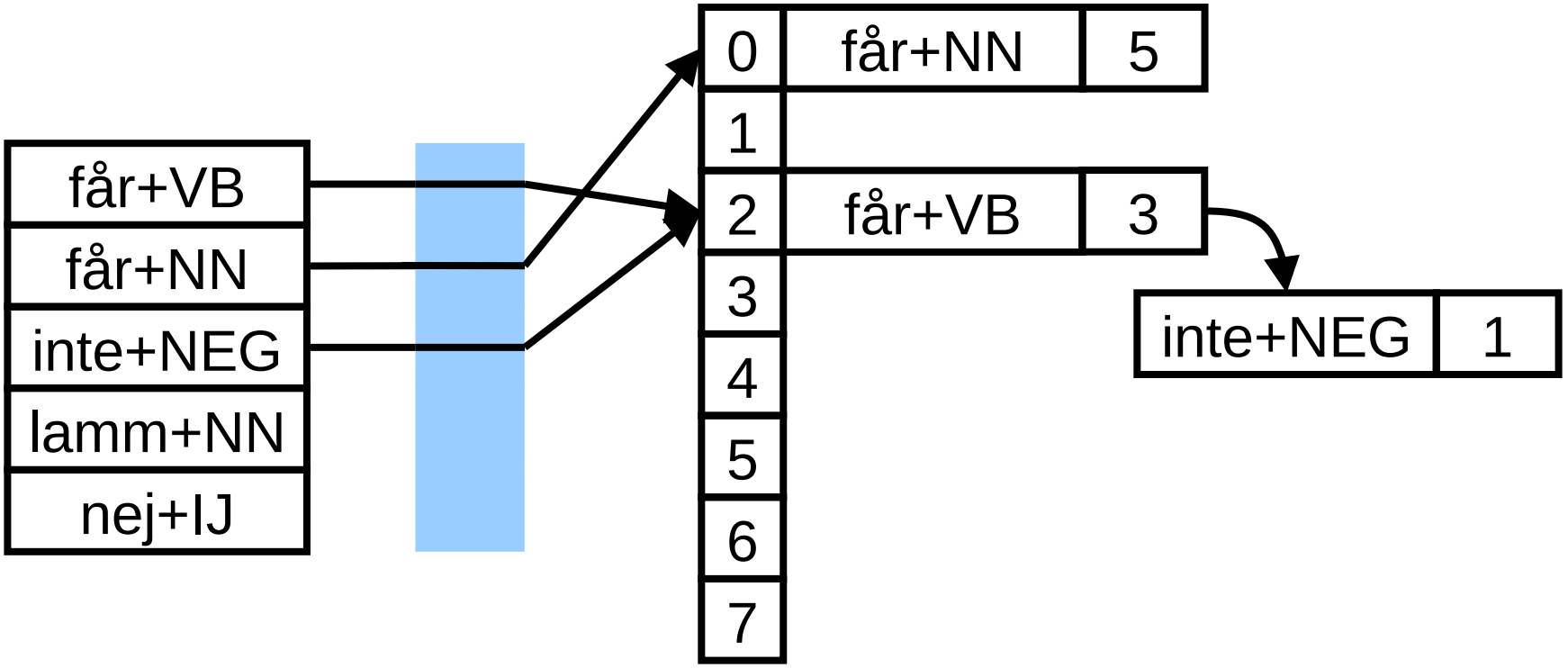
Hash tables: collisions



UPPSALA UNIVERSITET

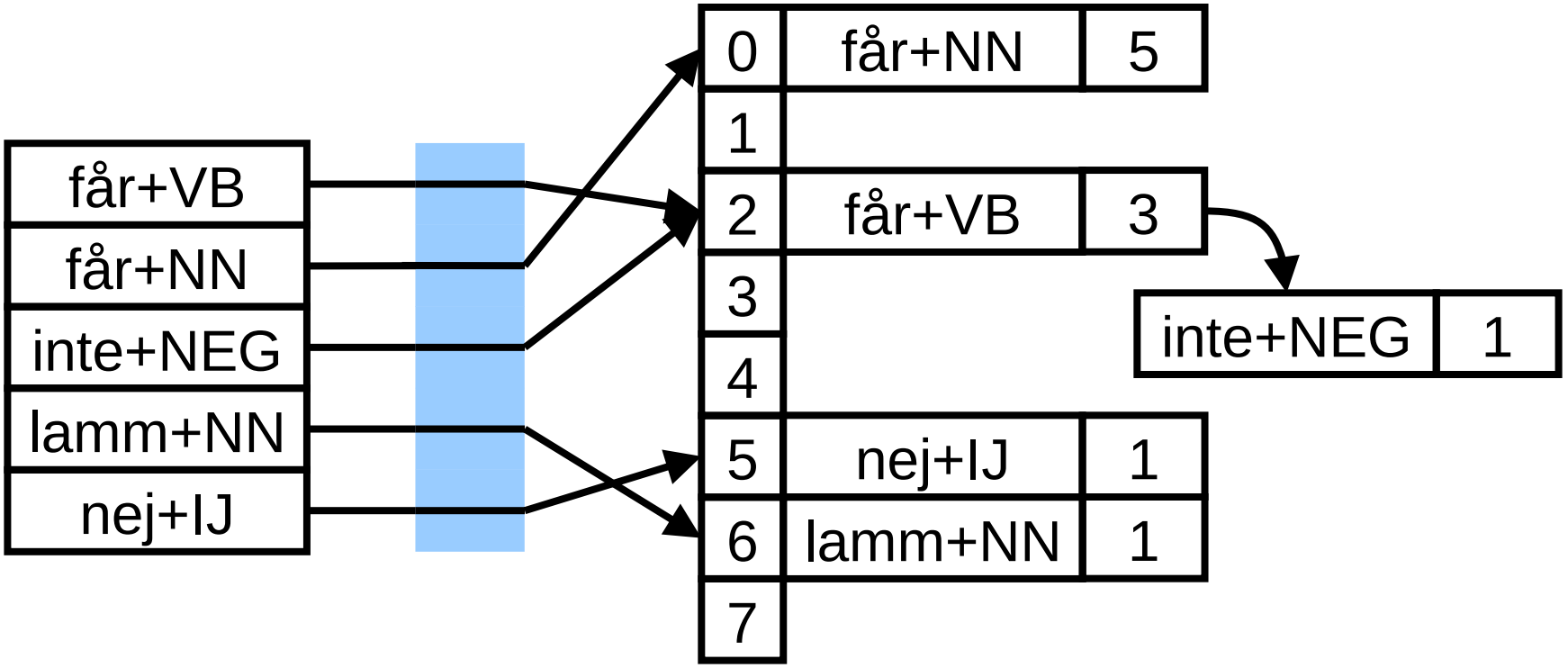
Hash tables: collisions

Linking



Hash tables: collisions

Linking



UPPSALA
UNIVERSITET

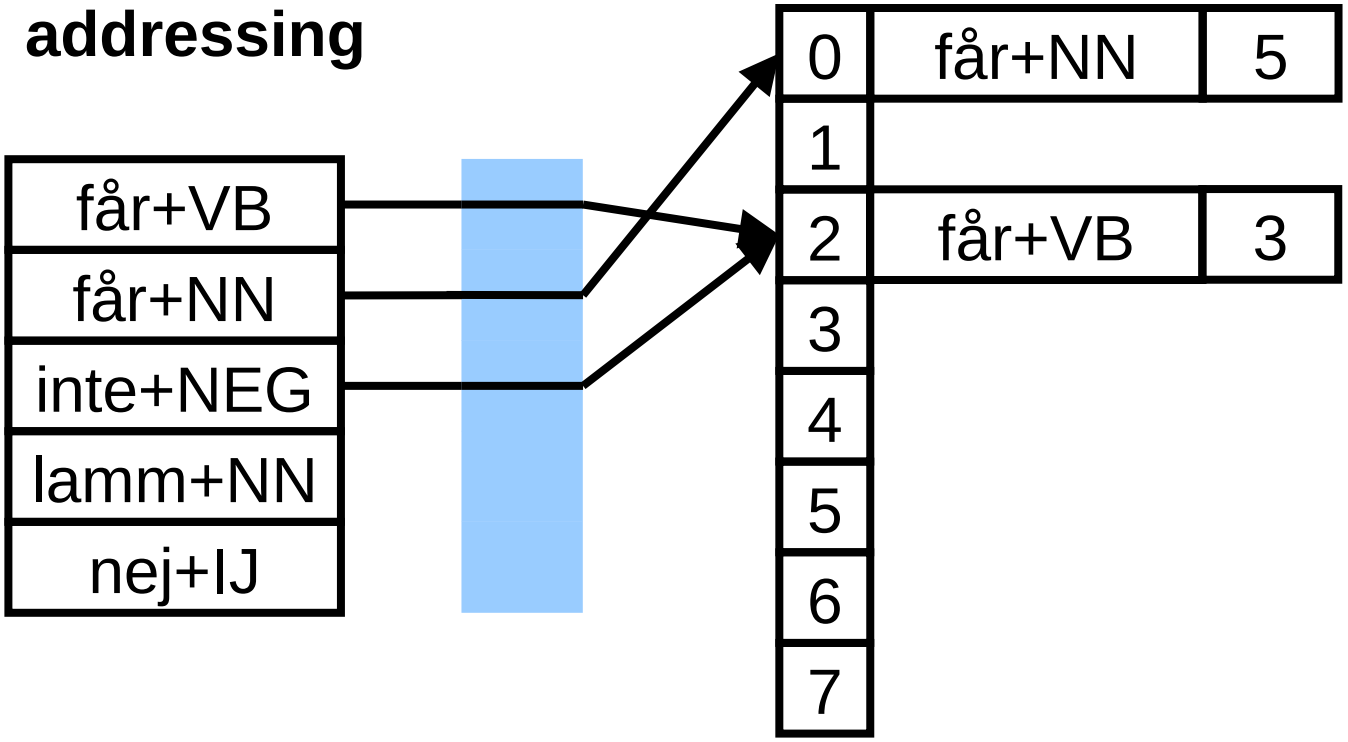
Linking

- Each position in the array is a linked list with key value pairs
- Search
  - Apply the hash function on the key
  - Go through the corresponding linked list until:
    - The key is found
    - You reach the end of the list (the key does not exist)

UPPSALA UNIVERSITET

Hash tables: collisions

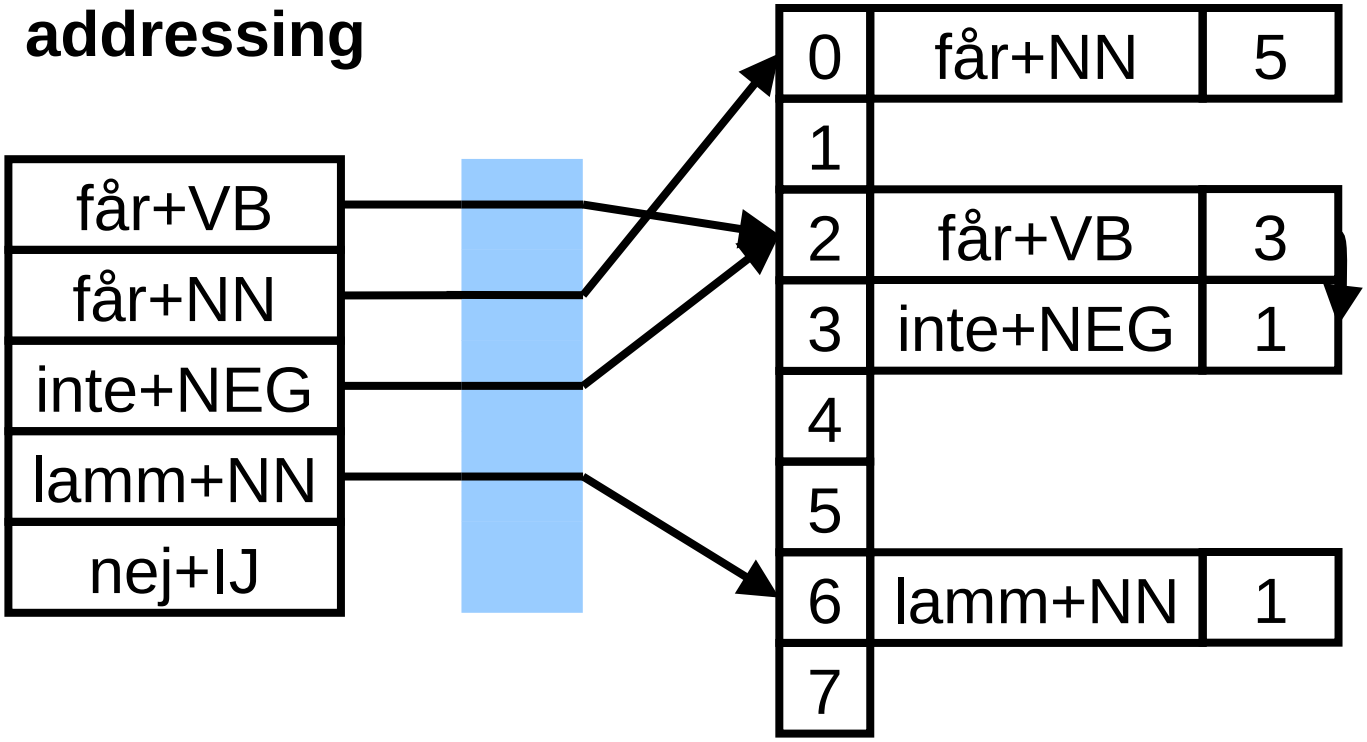
Open addressing



UPPSALA UNIVERSITET

Hash tables: collisions

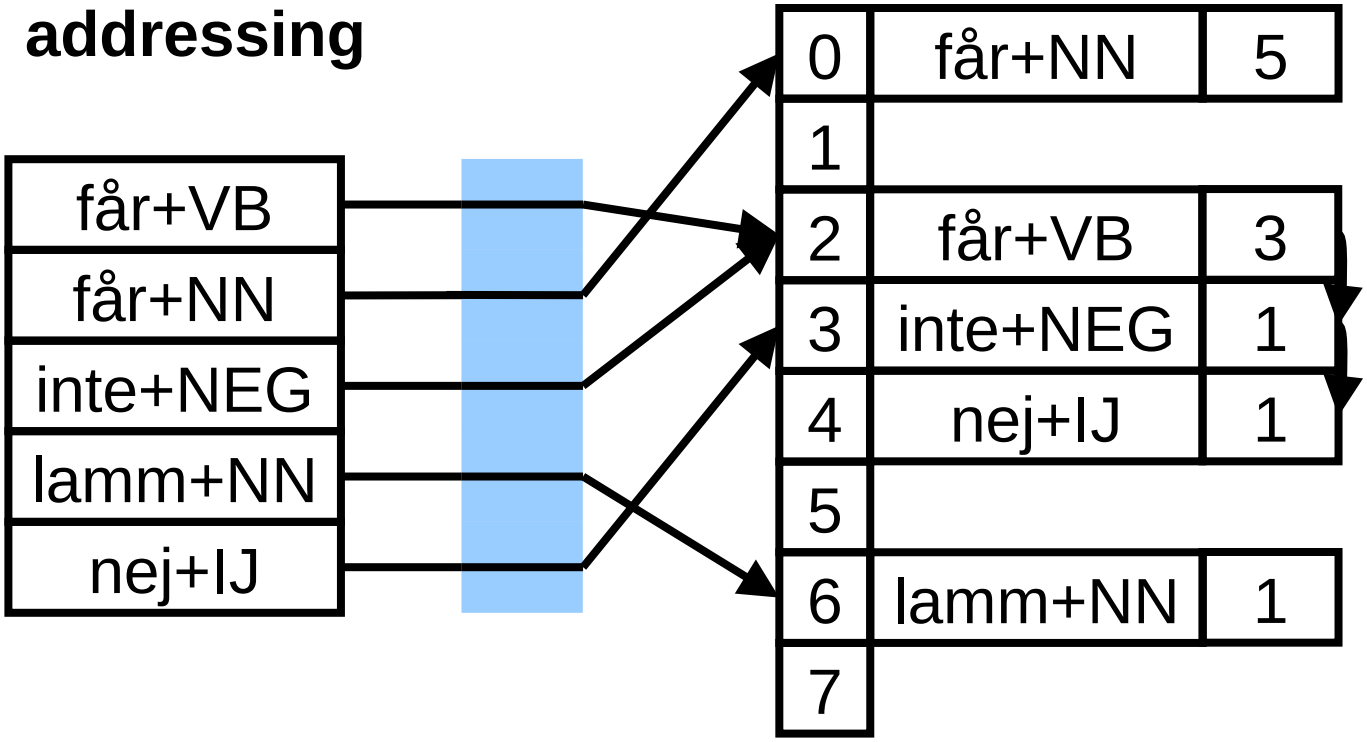
Open addressing



UPPSALA UNIVERSITET

Hash tables: collisions

Open addressing



UPPSALA
UNIVERSITET

Open addressing

- Search
  - Apply the hash function on the key
  - If the position is taken by another key:
    - Go to another position in the array
    - Repeat until the key or an empty position is found
- Different strategies
  - Go one step at the time
    - Linear probing
  - Increase the step length quadratically
    - Quadratic probing (1,2,4,...)
  - Let another hash function decide the step length
    - Double hashning

UPPSALA
UNIVERSITET

Load factor

- The higher the number of keys in a hash table, the higher the risk of collision
- The more collisions, the slower insertion/search
- Trade-off between time and space
- Load factor (λ): number of keys/size
  - 0,75 is often good
  - It depends on the data and application

UPPSALA
UNIVERSITET

Python dictionary

- The standard python dictionary is a hash table
- It uses open addressing to handle collision (with an advanced probing method)
- Size is automatically increased when $\lambda > 2/3$
  - By rehashing (expensive)

UPPSALA
UNIVERSITET

Complexity

	Best case	Worst case	Amortized
Insertion	O(1)	O(n)	O(1)
Search	O(1)	O(n)	O(1)
Deletion	O(1)	O(n)	O(1)

- What affects the worst case?
  - Resizing
- What effects the best time?
  - The load factor (Tradeoff time/space)
  - The method for collision handling
- The time for individual operations may vary considerable: O(1) or O(n) if resizing