Data Structures for Sets

- Many applications deal with sets.
  - Compilers have symbol tables (set of vars, classes)
  - Dictionary is a set of words.
  - Routers have sets of forwarding rules.
  - Web servers have set of clients, etc.
- A set is a collection of members
  - No repetition of members
  - Members themselves can be sets
- Examples
  - Set of first 5 natural numbers: {1,2,3,4,5}
  - {x | x is a positive integer and x < 100}
  - {x | x is a CA driver with > 10 years of driving experience and 0 accidents in the last 3 years}

Set Operations

Binary operations	Member	Set
Member	Order (=, <, >)	Find, insert, delete, split, …
Set	Find, insert, delete, split, …	Union, intersection, difference, equal, …

- Unary operation: min, max, sort, makenull, …

Observations

- Set + Operations define an ADT.
  - A set + insert, delete, find
  - A set + ordering
  - Multiple sets + union, insert, delete
  - Multiple sets + merge
  - Etc.
- Depending on type of members and choice of operations, different implementations can have different asymptotic complexity.

Set ADT: Union, Intersection, Difference

AbstractDataType ***SetUID***

instance
multiple sets

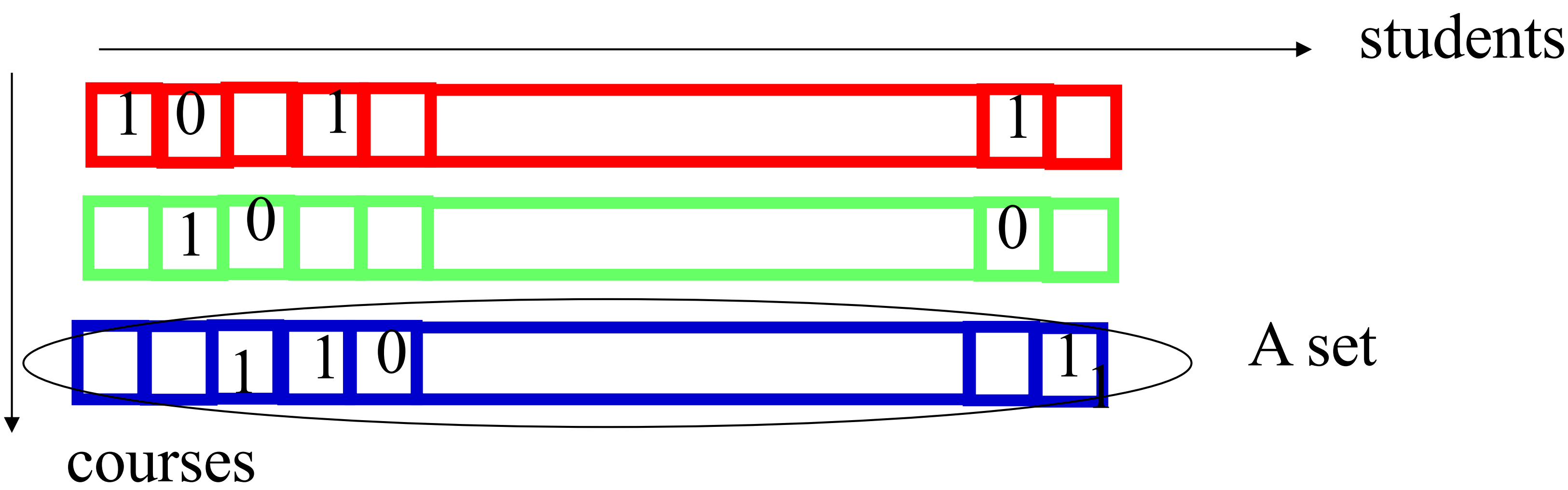
operations
union (***s1,s2***): {x | x in s1 *or* x in s2}
intersection (***s1,s21***): {x | x in s1 *and* x in s2}
difference (***s1,s2***): {x | x in s1 *and* x *not* in s2}

Examples

- Sets: Articles in Yahoo Science (A), Technology (B), and Sports (C)
  - Find all articles on Wright brothers.
  - Find all articles dealing with sports medicine
- Sets: Students in CS8 (A), CS16 (B), and CS40 (C)
  - Find all students enrolled in these courses
  - Find students registered for CS8 only
  - Find students registered for both CS8 and CS16
  - Etc.

Set UID Implementation: Bit Vector

- Set members known and finite (e.g., all students in CS dept)



- Operations
  - Union: u[k]= x[k] | y[k];
  - Intersection: u[k] = x[k] & y[k];
  - Difference: u[k] = x[k] & ~y[k];
- Complexity: O(n): *n* size of the set

Set UID Implementation: linked lists

- Bit vectors great when
  - Small sets
  - Known membership

- Linked lists
  - Unknown size and members
  - Two kinds: Sorted and Unsorted

Set UID Complexity: Unsorted Linked List

- Intersection

  For k=1 to n do

  Advance set_A one step to find kth element;

  Follow set_B to find that element in B;

  If found then

  Append element k to set_{AB}

  End

- Searching for each element can take n steps.
- Intersection worst-case time $O(n^2)$.

Set UID Complexity: Sorted Lists

- The list is sorted; larger elements are to the right
- Each list needs to be scanned only once.
- At each element: increment and possibly insert into A&B, constant time operation
- Hence, sorted list set-set ADT has O(n) complexity
- A simple example of how even trivial algorithms can make a big difference in runtime complexity.

Set UID: Sorted List Intersection

- Case A $*set_A=*set_B$
  - Include $*set_A$ (or $*set_B$) in $*set_{AB}$
  - Increment set_A
  - Increment set_B
- Case B $*set_A<*set_B$
  - Increment set_A Until
  - $*set_A=*set_B$ (A)
  - $*set_A>*set_B$ (C)
  - $*set_A==null$
- Case C $*set_A>*set_B$
  - Increment set_B Until
  - $*set_A=*set_B$ (A)
  - $*set_A<*set_B$ (B)
  - $*set_B==null$
- Case D $*set_A==null$ or $*set_B==null$
  - terminate

Dictionary ADTs

- Maintain a set of items with distinct keys with:
  - *find* (k): find item with key k
  - *insert* (x): insert item x into the dictionary
  - *remove* (k): delete item with key k
- Where do we use them:
  - Symbol tables for compiler
  - Customer records (access by name)
  - Games (positions, configurations)
  - Spell checkers
  - Peer to Peer systems (access songs by name), etc.

Naïve Implementations

- The simplest possible scheme to implement a dictionary is "log file" or "audit trail".
  - Maintain the elements in a linked list, with insertions occuring at the head.
  - The search and delete operations require searching the entire list in the worst-case.
  - Insertion is O(1), but find and delete are O(n).
- A sorted array does not help, even with ordered keys. The search becomes fast, but insert/delete take O(n).