

CI2612: Algoritmos y Estructuras de Datos II

Blai Bonet

Universidad Simón Bolívar, Caracas, Venezuela

Conjuntos dinámicos

© 2017 Blai Bonet

Objetivos

- Introducir el concepto de conjunto dinámico y las operaciones que deben ser soportadas
- Resumen de diferentes implementaciones y su desempeño

© 2017 Blai Bonet

Conjuntos dinámicos

La representación eficiente de conjuntos de elementos es una tarea fundamental para la implementación de algoritmos avanzados

A lo largo de la ejecución de un programa, el contenido del conjunto puede cambiar a medida que elementos son insertados y eliminados. Adicionalmente el algoritmo requiere de al menos poder chequear cuando un elemento pertenece o no al conjunto

Un conjunto que cambia se conoce como **conjunto dinámico**. En el resto del curso nos enfocaremos en como representar dichos conjuntos

© 2017 Blai Bonet

Elementos del conjunto dinámico

Típicamente los elementos son objetos cuyos atributos pueden ser accedidos/modificados si se tiene un **apuntador/referencia** al objeto

Frecuentemente se asume que los objetos tienen **claves** distintas y un conjunto dinámico puede ser entendido como la colección de las claves de los objetos en el conjunto

Cuando tratamos a los objetos como claves, los atributos distintos a la clave se llaman **datos satélite**

En algunos casos asumiremos que existe un **orden total sobre las claves** de forma que claves distintas puedan ser comparadas

Operaciones sobre el conjunto dinámico

Una estructura de datos (ED) que implementa un conjunto dinámico esta diseñada para **soportar de forma eficiente** un número de operaciones sobre el conjunto

Las operaciones se dividen en dos tipos:

- operaciones tipo **"query"** que devuelven información sobre los elementos y el conjunto pero que no lo modifican
- operaciones que **modifican la ED**

Operaciones tipo query

- **Search(S, k)**: busca un elemento con clave k . Si lo encuentra, retorna un apuntador x tal que $x.key=k$, sino retornal **null**
- **Minimum(S)**: para claves ordenadas, retorna un apuntador al elemento en S de menor clave
- **Maximum(S)**: para claves ordenadas, retorna un apuntador al elemento en S de mayor clave
- **Successor(S, x)**: para claves ordenadas, retorna un apuntor al elemento en S cuya clave es la siguiente a la clave de x , o **null** si no existe tal elemento
- **Predecessor(S, x)**: para claves ordenadas, retorna un apuntador al elemento en S cuya clave es la anterior a la clave de x , o **null** si no existe tal elemento

Operaciones que modifican la ED

- **Insert(S, x)**: inserta el elemento apuntado por x en el conjunto S
- **Delete(S, x)**: elimina del conjunto S el elemento apuntado por x (observe que x es un apuntador y no una clave)

Una ED que soporta **Search(S, k)**, **Insert(S, x)** y **Delete(S, x)** se conoce como un **diccionario de datos**

Diversas EDs para conjuntos dinámicos

En lo que resta del curso veremos diferentes EDs que implementan conjuntos dinámicos:

- EDs elementales: pilas, colas, listas enlazadas, y la implementación de objetos y apuntadores cuando el lenguaje de programación no lo soporta
- tablas de hash que implementan un diccionario y cuyas operaciones pueden realizarse en **tiempo promedio constante** bajo suposiciones razonables
- árboles de búsqueda binarios
- árboles rojo/negro que soportan todas las operaciones anteriores en **tiempo logarítmico** en el número de elementos