Algorithms for Problem Solving – 11650

Strings, Hash Tables and Sorting Algorithms

Jon Ander Gómez Adrián
jon@dsic.upv.es

Departament de Sistemes Informàtics i Computació
Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

4 de febrero de 2014

Strings

Representaciones internas para strings:

- Vectores o arrays con terminador (C/C++)
  - Utiliza menos memoria
- Vector más longitud
  - Limita la longitud según implementación
  - ¿ `char str[300]` ?
- Lista enlazada
  - Operaciones de inserción/borrado más rápidas

Strings: Búsqueda directa

- Busca el patrón o cadena a encontrar a partir de cada posición en el texto
- Son dos bucles anidados
- $T(n) \in O(|s| * |p|)$
  - s es la string con el texto donde buscar
  - p es el patrón o subcadena a buscar
- ¿Existe una estrategia más eficiente?

Strings: Búsqueda de Boyer-Moore

- Utiliza una estrategia que no intenta encontrar el patrón a partir de todas las posiciones del texto
- Aunque también hay un bucle dentro de otro, el índice sobre el texto se incrementa a saltos según una tabla de distancias precalculada
- El carácter o símbolo del alfabeto en el texto se utiliza como índice para la tabla de distancias
- El índice sobre el texto avanza de manera ascendente
- La concordancia se busca de manera descendente
- $T(n) \in O(|s|)$
  - s es la string con el texto donde buscar

Estructuras de datos

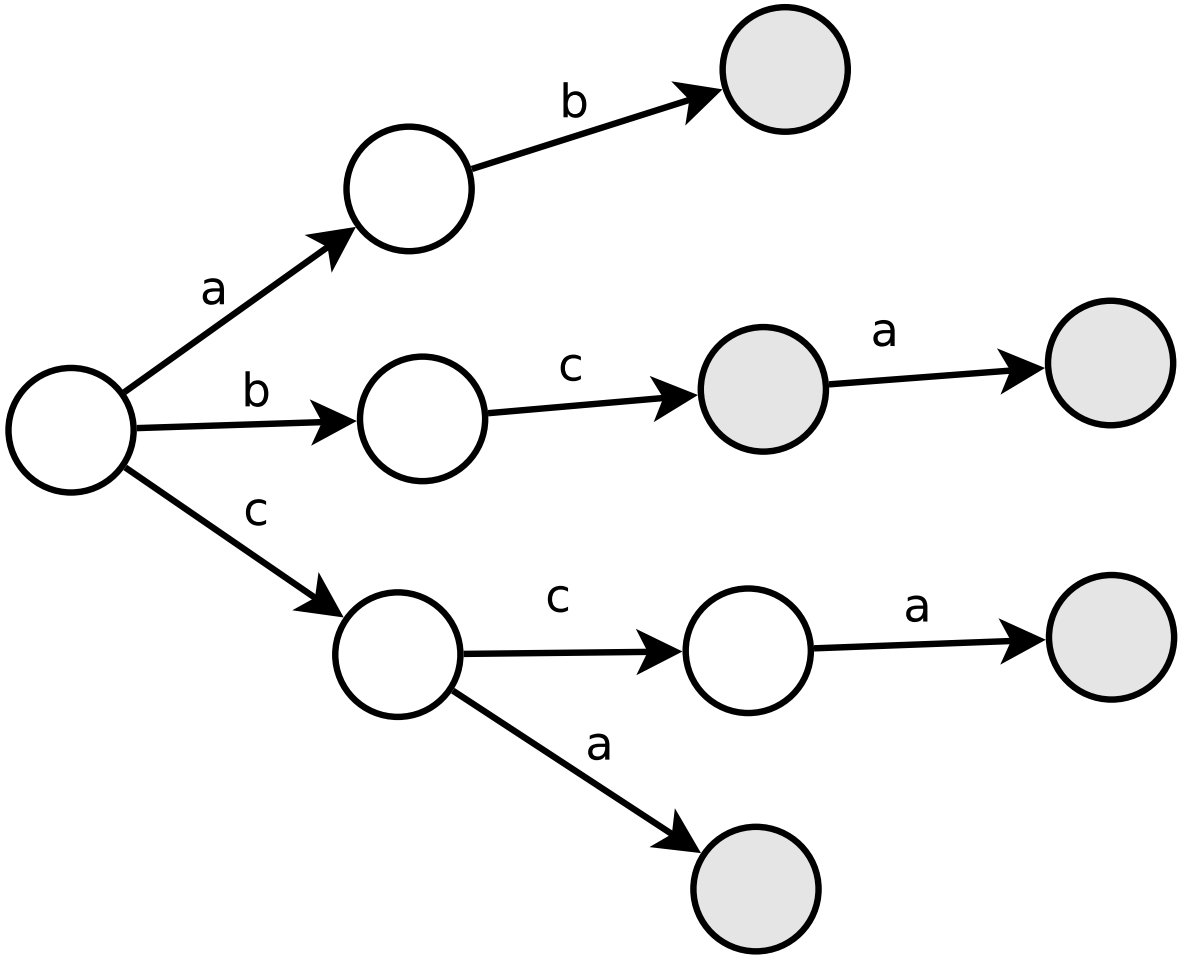
	C++ STL	Java
Pilas	stack<int>	Stack<Integer>
Colas	queue<int>	List<Integer>
Colas con prioridad	priority_queue<int>	SortedMap<Integer>
Diccionarios	map<int> hash_map<int>	Hashtable<Integer> HashMap<Integer>
Conjuntos	set<int,lstr>	HashSet<Integer>

Tablas Hash

- Función hash que disperse lo mejor posible los objetos a almacenar
- La posición en la tabla será: `hash % HASH_SIZE`
- `HASH_SIZE` debe ser un número primo
- Veamos una implementación de función hash para strings obtenida de StackOverflow

  `http://stackoverflow.com/questions/114085/fast-string-hashing-algorithm-with-low-collision-rates-with-32-bit-integer`

Trie: Árbol aceptor de prefijos

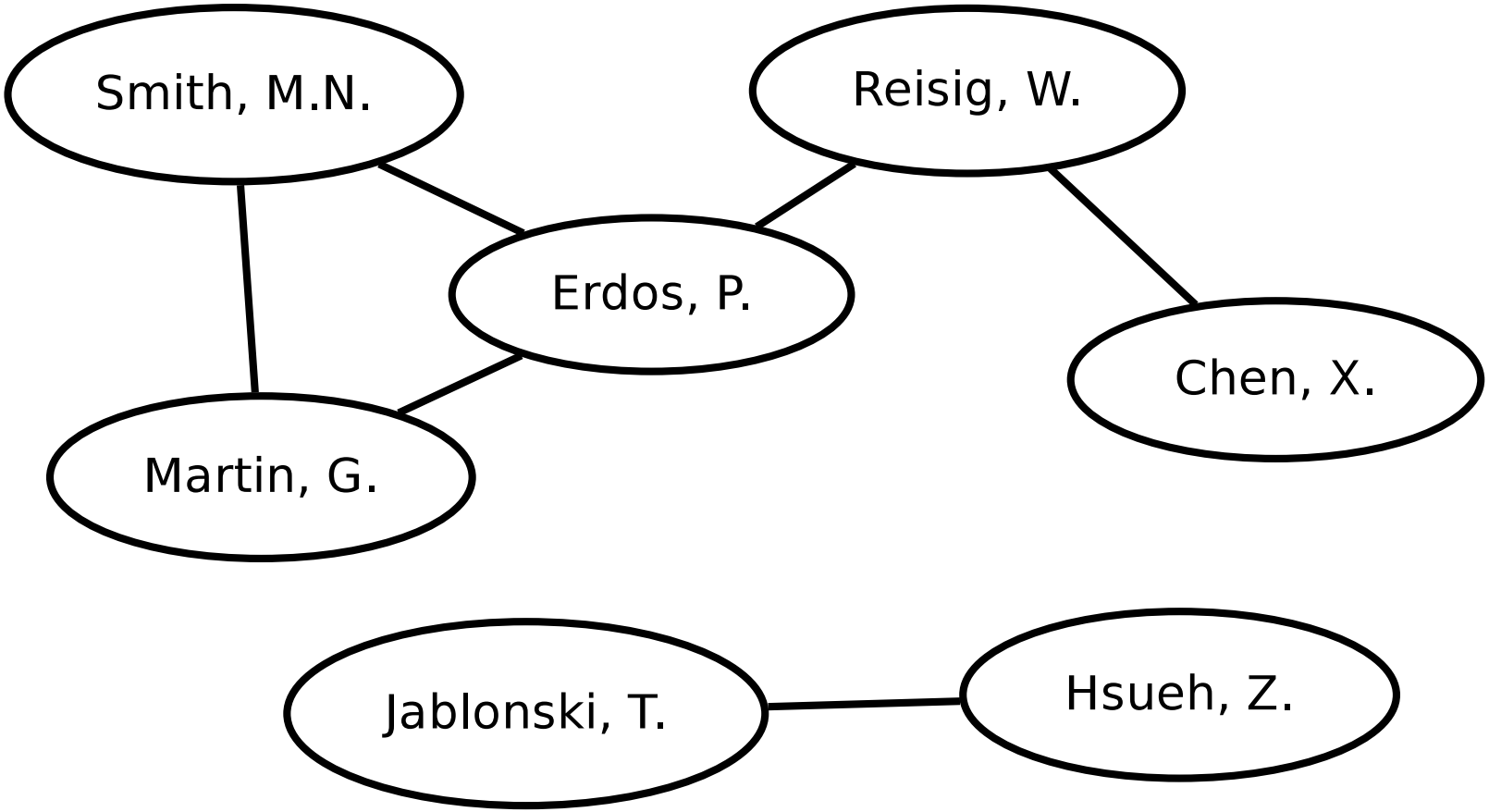


Ver código en `erdos2010-sin-solucion.java`
métodos `add()` y `get()` de la clase `Diccionario`

Erdös numbers

- Es necesario disponer los nombres de los autores en un diccionario para realizar las comprobaciones de manera eficiente
- Podemos utilizar una tabla Hash o un diccionario
- Cada autor debe ser un nodo de un grafo, el diccionario de autores cuya clave de búsqueda es el nombre
- De los datos de entrada podemos ignorar los títulos de los artículos
- El nombre de cada autor tiene dos partes separadas por una coma ...
- ... salvo el último de cada artículo, que puede no tener coma porque los dos puntos delimitan al identificador

Erdös numbers: grafo autores inicial



Erdös numbers: grafo autores final