11 Dizionari

Definizione 11.1. *Un dizionario è una* ***struttura dati astratta*** *e consiste in una collezione di* ***coppie*** *della forma:*

- ***Chiave****: è il meccanismo di accesso all'elemento ed è* ***univoca*** *nella collazione*
- ***Elemento****: è un qualsiasi tipo*

Le operazioni che possono essere eseguite sui dizionari sono *inserimento*, *ricerca* e *cancellazione*.

11.1 Indirizzamento diretto

Ogni elemento del dizionario viene mappato con una chiave estratta da un determinato universo U e che corrisponde alla posizione nell'array dove viene inserito.
La complessità di questa tecnica è $O(1)$ per tutte e tre le operazioni possibili sui dizionari. Il problema si presenta quando le chiavi usate N sono molto minori dell'universo iniziale U e abbiamo quindi che $\frac{N}{U} < 1$ con un conseguente **spreco di spazio di memoria** in quanto lo spazio nell'array dovrà comunque essere allocato.

11.2 Chaining

Un modo per evitare i problemi dell'*indirizzamento diretto* è utilizzando un array più piccolo dell'universo possibile di chiavi. Questo ovviamente causa situazioni in cui c'è un **conflitto** perché due o più elementi voglio utilizzare la stessa chiave. Viene quindi creata una **lista** di elementi associata alla posizione in cui si è creato il conflitto e ogni nuovo elemento che vuole inserirsi lì verrà messo sulla testa della lista.
Per capire ogni elemento a quale chiave deve essere associato si utilizza una **funzione di hash** che deve essere definita in modo da ridurre il numero di collisioni e da generare sempre lo stesso indice per la stessa chiave.

11.3 Open addressing

I conflitti di indirizzo vengono risolti cercando la prima posizione libera. Questo processo di ricerca viene chiamato **probing**. Questa operazione può avvenire in diversi modi:

- *Lineare*: vado alla posizione successiva
- *Quadratico*: uso una funzione quadratica
- *Doppio hash*: utilizzo la funzione hash per cercare la posizione successiva

Esempio 11.1. Esempio di esecuzione di *probing*:

$$\begin{gathered} probe(0) \longrightarrow h(i) \\ probe(1) \longrightarrow (h(i) + f(1)) \\ \vdots \\ probe(j) \longrightarrow (h(i) + f(j)) \\ \vdots \\ probe(m-1) \longrightarrow Fail \end{gathered}$$

Esempio 11.2. Esempio di *probing lineare*.

Osservazione 11.3.1. Il problema che nasce dell'open addressing lineare è il **clustering primario**. Quando infatti si verifica una collisione, viene riempita una cella di memoria nelle vicinanze, aumentando così il rischio di collisione e aumentando il tempo necessario per la ricerca.
Questo avviene perché la *probabilità* che uno slot preceduto da slot pieni sia riempito è pari a $\frac{i+1}{m}$.

Osservazione 11.3.2. Il problema che deriva dall'utilizzo di open addressing quadratico è il **clustering secondario**. Infatti in questo caso se due chiavi hanno la stessa posizione iniziale la loro sequenza di tentativi sarà la stessa.