Algorismes i Estructures de Dades II. Programa

1. Introducció
2. Arbres de cerca
3. Dispersió
4. Selecció
5. Estructures específiques

Estructures de Dades

- La solució eficient a certs problemes depén de la manera en què es gestione i s'organitze la informació.
- Aquesta gestió i organització implica algorismes d'accés i modificació de les dades emmagatzemades.
- Entenem com a estructura de dades una forma particular d'emmagatzemar informació així com les diferents maneres (algorismes) d'accedir-hi.
- Algunes estructures de dades (i els corresponents algorismes de manipulació) poden arribar a ser extremadament complexes.
- Un tipus abstracte de dades és una especificació d'una estructura de dades en la que es defineixen les operacions i el seu efecte de forma independent de la implementació (tant de l'estructura com dels algorismes).

Tipus de Dades Abstractes

- L'abstracció és una característica fonamental del llenguatges de programació moderns.
- De fet, aquest és un tret fonamental en el paradigma de l'orientació a objectes.
- Des del punt de vista del programador, l'ús d'abstraccions permet resoldre problemes complexos sense haver d'implementar estructures de dades i algorismes de manipulació.
- Aquestes estructures es poden trobar implementades en diferents biblioteques (estandar o no).
- No obstant això, és important conéixer la forma en què les més importants d'aquestes han sigut implementades i per què. D'aquesta manera, es pot triar entre diferents alternatives i decidir si val la pena o no re-implementar alguna d'aquestes per a alguna aplicació particular.

Objectiu de l'assignatura

- L'objectiu de l'assignatura és mostrar algunes de les més importants estructures de dades (no bàsiques).
- Normalment, es tracta de famílies d'estructures que han anat cresquent a mesura que han sigut proposades.
- En molts casos, no es pot dir que una alternativa és millor que un altra perque això pot dependre de l'ús concret a l'hora de resoldre un problema.
- Aprendrem també com es poden avaluar els diferents aspectes de les estructures de dades i com decidir una vegada conegudes les característiques de cada problema.

Exemple (TAD Cua. Interfície)

```
package DataStructures;
import Exceptions.*;

public interface Queue
{
    boolean isEmpty( );

    Object getFront( ) throws Underflow;

    Object dequeue( ) throws Underflow;

    void  enqueue( Object X );

    void makeEmpty( );
}
```

Exemple (TAD Cua. Implementació)

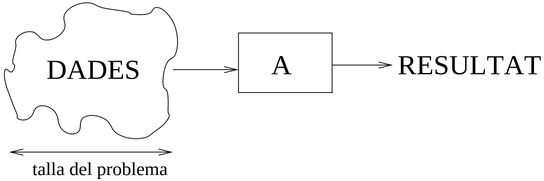
```
package DataStructures;
import Exceptions.*;

public class QueueAr implements Queue
{   public QueueAr( )
    {   theArray = new Object[ DEFAULT_CAPACITY ];
        makeEmpty( );
    }
    public boolean isEmpty( )
     ...
    private int increment( int x )
     ...
    private Object [ ] theArray;
    private int        currentSize;
    private int        front;
    private int        back;

    static final int DEFAULT_CAPACITY = 10;
}
```

Exemple (TAD Cua. Ús)

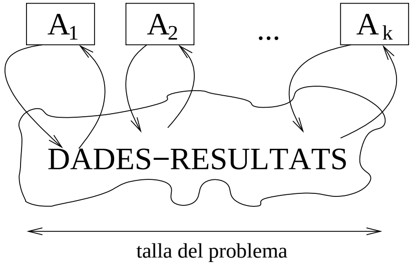
```
import DataStructures.*;
import Exceptions.*;

public final class TestQueue
{
    public static void main( String [ ] args )
    {
        Queue q = new QueueAr( );
        for( int i = 0; i < 5; i++ )
            q.enqueue( new Integer( i ) );

        System.out.print( "Contents:" );
        try
        { for( ; ; ) System.out.print( " " + q.dequeue( ) ); }
        catch( Underflow e ) { }

        System.out.println( );
    }
}
```

Anàlisi

En analitzar un algorisme, es considera una certa entrada o instància (de talla n) i es caracteritza el cost de resoldre-la.



En canvi, quan implementem un TAD, tenim un conjunt d'operacions (mètodes) que prenen com a entrada i modifiquen l'estructura de suport de les dades.



Anàlisi Amortitzada

En general, el cost de cada operació no és independent de les altres.

En lloc de caracteritzar el cost (millor, pitjor, mitjà) de cada operació cal considerar **seqüències** d'operacions arbitràries sobre el TAD.

És possible que una operació tinga un cost pitjor elevat, però que aquest siga molt improbable quan aquesta s'aplica dins de seqüències d'operacions vàlides.

Anàlisi Amortitzada

Cost amortitzat: cost que avalua l'eficiència d'un conjunt d'operacions que s'apliquen en un mateix context.

Si no s'especifica d'altra manera, el cost amortitzat és una fita superior (cas pitjor). Diem que una operació té un cost amortitzat t, si en qualsevol seqüència d'operacions, la contribució total de la operació dividida per el nombre d'aquestes és menor o igual t.

Es pot considerar un cost amortitzat per a cada operació específica o per a totes les operacions.

Exemple. Piles amb extracció múltiple

Considerem el TAD pila enriquit amb una operació que extraga de la pila k elements.

$\texttt{buida : pila} \longrightarrow \texttt{bool}$
$\texttt{cim : pila} \longrightarrow \texttt{elem}$
$\texttt{apilar : pila} \times \texttt{elem} \longrightarrow \texttt{pila}$
$\texttt{desapilar : pila} \longrightarrow \texttt{pila}$
$\texttt{extraure : pila} \times \texttt{enter} \longrightarrow \texttt{pila}$

Amb qualsevol implementació lògica s'obtenen costs constants per a totes les operacions llevat de extraure que tindrà un cost proporcional al nombre d'elements extrets.

Exemple. Piles amb extracció múltiple

Si considerem seqüències qualssevol de m operacions, una fita superior evident del seu cost és m vegades el cost de la pitjor operació (lineal).

Però ..., pot haver **moltes** operacions **cares** en una mateixa seqüència?

Cas pitjor (A=apilar, E= extraure):

$$\underbrace{AA\ldots A}_{m-1} E(m-1)$$

Cost total: $2m-2$

Cost per operació: $2+O(\frac{1}{m})$

Exemple. Piles amb extracció múltiple

Es pot assignar un cost amortitzat diferent a cada operació.

Si no considerem el cost constant associat a l'intent d'extraure un o més elements de la pila quan aquesta està buida, podem raonar de la següent manera:

> Les operacions `desapilar` i `extraure`, només poden contribuir al cost si efectivament extrauen elements i la seua contribució serà exactament igual al nombre d'aquests.

Només hi haurà elements per extraure si han sigut prèviament apilats.

Exemple. Piles amb extracció múltiple

Si per cada element que apilem, comptem una unitat de cost extra, estarem tenint en compte la possibilitat de que aquest element puga ser extret més endavant.

	millor	pitjor	amortitzat
`apilar`	1	1	2
`desapilar`	0	1	0
`extraure`	0	k	0

Cost Amortitzat. Definició

Cost de cada operació, dins d'un conjunt d'aquestes aplicades en un mateix context de forma que per a tota seqüència d'operacions, la suma dels costs amortitzats és una fita superior del cost real de la seqüència.

El cost amortitzat és el que cal conéixer. Les operacions es comporten exactament igual (o millor) que si el seu cost real fóra l'amortitzat.

Exemple. Vectors extensibles

En la implementació estàtica de piles i cues cal definir una grandària màxima del vector subjacent.

Si se supera aquesta, es pot llançar una excepció o bé redimensionar (internament) el vector per tal d'admetre més elements.

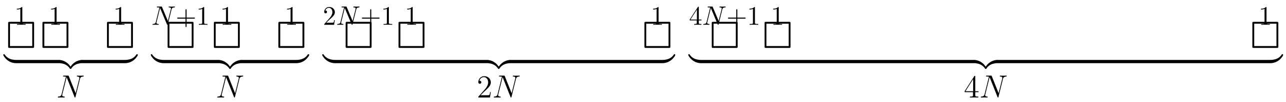
La pràctica habitual consisteix en definir un nou vector del doble de grandària que l'actual i copiar els elements.

En el cas de la pila, això modificaría el cost *real* de `apilar` que tindria un cas pitjor **lineal**!

Què passa amb el cost amortitzat?

Exemple. Vectors extensibles

Considerem només l'operació `apilar` que costaria 1 llevat del cas en que calga duplicar la grandària. Siga N la talla inicial.

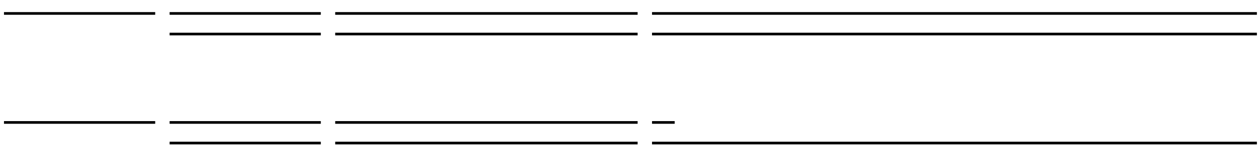


Hi ha operacions que tenen costs $2N + 1$, $4N + 1$, etc. Aquests costs es poden veure com que ja s'està pagant una unitat de cost tant si les noves posicions s'ocupen com si no.

Exemple. Vectors extensibles

En el moment que totes les posicions estan ocupades (millor), el cost haurà sigut de 2 unitats per posició (menys N).

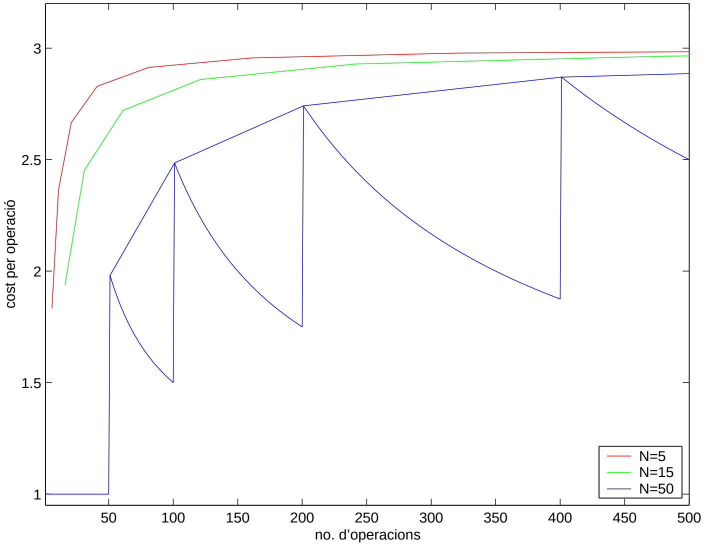
En el moment que s'acaba de duplicar la grandària a $2N'$ (pitjor), s'haurà tingut un cost de N' associat a les noves posicions que és com si les N' (més una) operacions anteriors hagueren costat 3 en lloc de 2.



$\boxed{\text{La operació } \texttt{apilar} \text{ té un cost amortitzat de 3.}}$

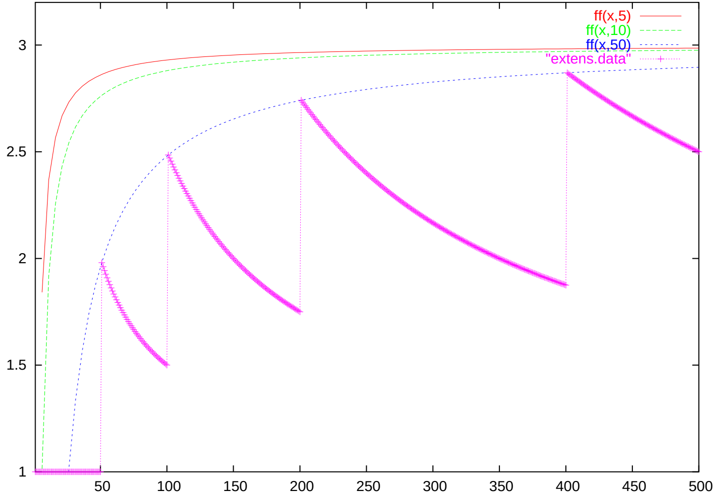
Exemple. Vectors extensibles

Si representem el cost per operació a mesura que creixen el nombre d'operacions apilar s'obté:



Exemple. Vectors extensibles

També es poden representar les mateixes dades però juntament amb una funció fita superior "suau" obtinguda analíticament:



Mètode del Potencial

És una manera de formalitzar l'obtenció del cost amortitzat.

No hi ha un únic cost amortitzat. Tot conjunt de costs que garantesca una fita superior per a tota seqüència és un cost amortitzat. Aquesta fita pot ser ajustada o no.

Considerem que un conjunt d'operacions actua i modifica una certa estructura de dades de suport que canvia d'estat com a conseqüència de cada operació.

Potencial: Qualsevol valor numèric que caracteritza cada un dels estats d'aquesta estructura.

Mètode del Potencial

Qualsevol operació produeix un **increment** de potencial que pot ser positiu, negatiu o zero i tota seqüència d'operacions dóna lloc a una seqüència de valors per al potencial $\phi_0, \phi_1, \ldots, \phi_m$

Si el potencial inicial és menor o igual que qualsevol altre ($\phi_0 \leq \phi_k \forall k > 0$) aleshores el cost amortitzat per a l'operació i-èssima es pot definir com a

$$\boxed{\hat{c}_i = c_i + \phi_i - \phi_{i-1}}$$

Ja que $\sum_i \hat{c}_i = \sum_i c_i + \underbrace{\phi_m - \phi_0}_{\geq 0} \geq \sum_i c_i$

Qualsevol fita superior de $\hat{c}_i$ també serà un cost amortitzat!

Exemple. Pila amb extracció múltiple

Potencial: nombre d'elements en la pila.

Potencial inicial: 0 (qualsevol altre és major o igual)

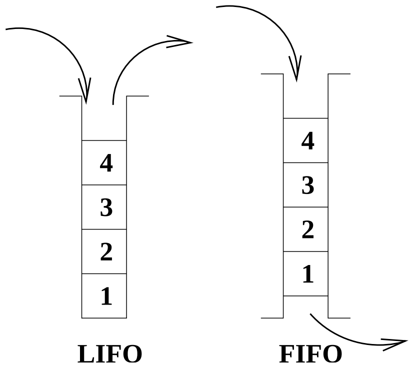
`apilar:` $\hat{c}_i = 1 + 1 = 2$

`despilar:` $\hat{c}_i = \max(0 + 0, 1 - 1) = 0$

`extraure:` $\hat{c}_i = \max(0 + 0, k - k) = 0$

Per tant, 2, 0 i 0 són fites superiors (ajustades) de $\hat{c}_i$ i es poden considerar com a costs amortitzats de les operacions.

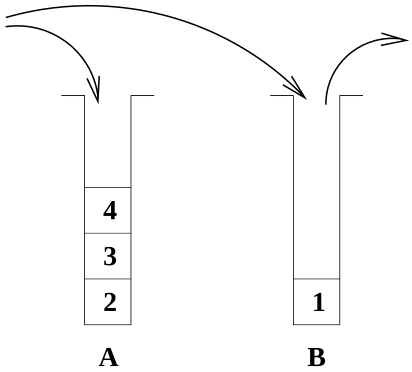
Exemple. Piles i cues



Es pot implementar una cua mijantçant piles?

Exemple. Piles i cues

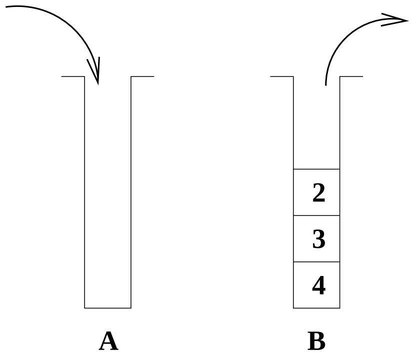
Considerem dues piles, A i B



La pila B contindrà sempre al seu cim el primer element.

Exemple. Piles i cues

Què passarà quan desencuem el primer element?



Aleshores es pot passar el tot el contingut de la pila A a la B amb la qual cosa el segon element quedarà al cim. (I a més a més podrem extraure fàcilment els següents).

Exemple. Piles i cues

encuar(x): $O(1)$

```
si buida(B) aleshores apilar(B,x)
si no apilar(A,x)
```

desencuar(x): $O(k)$

```
si !buida(B) aleshores
    desapilar(B)
    si buida(B) aleshores
        mentre !buida(A)
            apilar(B,desapilar(A))
```

Exemple. Piles i cues

Anàlisi amortitzada.

Potencial: $\phi_i =$ nombre d'elements en la pila B.

$$\texttt{encuar} \begin{cases} \text{millor:} & \hat{c}_i = 1 + 0 = 1 \\ \text{pitjor:} & \hat{c}_i = 1 + 1 = 2 \end{cases}$$

$$\texttt{desencuar} \begin{cases} \text{millor:} & \hat{c}_i = 1 + (-1) = 0 \\ \text{pitjor:} & \hat{c}_i = k + 1 + (k - 1) = 2k \end{cases}$$

s'obté un cost amortitzat (fita superior) lineal!!

$\boxed{\text{serà ajustada?}}$

Exemple. Piles i cues

Potencial: ϕ_i = nombre d'elements en la pila A.

$$\texttt{encuar} \begin{cases} \text{millor:} & \hat{c}_i = 1 + 1 = 2 \\ \text{pitjor:} & \hat{c}_i = 1 + 0 = 1 \end{cases}$$

$$\texttt{desencuar} \begin{cases} \text{millor:} & \hat{c}_i = 1 + 0 = 1 \\ \text{pitjor:} & \hat{c}_i = k + 1 + (-k) = 1 \end{cases}$$

El cost amortitzat de les dues operacions és constant!