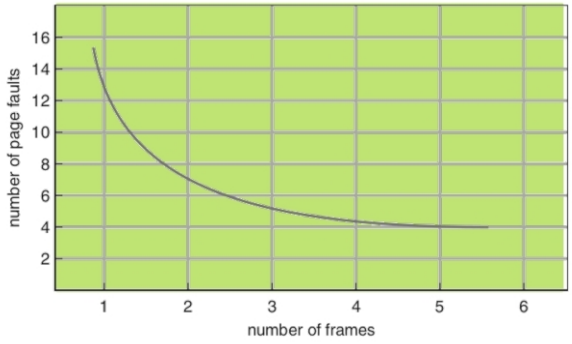
Considerazioni dell'algoritmo di Aging

Come si confrontano gli algoritmi fra loro? Si prende in considerazione la seguente metrica: il ***NUMERO DI PAGE FAULT*** che la politica dell'algoritmo genera, **rispetto all'algoritmo ottimale** oppure **rispetto a diversi algoritmi**

- Si suppone di avere una memoria suddivisa in **3 frame**. Questo numero determina l'efficienza di un sistema. Teoricamente, più frame ci sono, meno è la probabilità di avere page fault
- 

- Si suppone che si ha la sequenza di referenziazione: 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1
  - Cioè si referenzia la pagina 7, poi 0, poi 1 ecc..

Si referenzia la pagina 7:

- Inizialmente si ha un page fault perchè niente è caricato in memoria.
- Quindi idem per la pagina 0 e per la 1
- Si generano 3 page fault obbligatori
- 

SI referenzia la pagina 2:

- Quella chimata in causa più lontana è la 7 e viene sostituita con 2

L'algoritmo ottimale deve rimuovere quella che, vedendo la sequenza, viene chiamata più lontano fra quelle disponibili:

Quindi l'algoritmo genera 9 fault di pagina complessivi

Come detto prima, l'algoritmo ottimale non è realizzabile. Di conseguenza risulta che più ci si avvicina a 9, migliore è l'algoritmo..

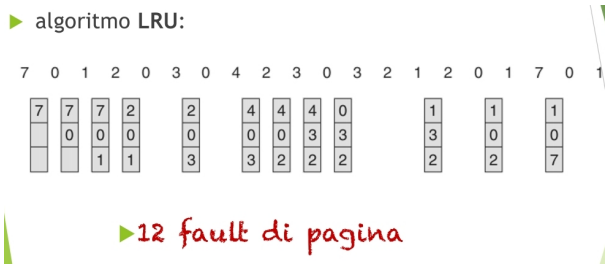
Valutazione Algoritmo FIFO

- Ogni volta che avviene un page fault, viene fatta una `dequeue()` quindi viene rimossa la testa della coda e viene sostituita.
- Vengono fatte le seguenti operazioni: `dequeue()` seguita da una `enqueue()`

> *NOTA: Quando si fa una* `dequeue()` *la testa avanza di 1*, quindi la coda si modifica ad ogni page fault

L'algoritmo FIFO ha generato 15 page fault.

Valutazione algoritmo LRU



- Sostituisce la pagina meno usata di frequente fra le presenti.
- In coda si avranno quelle pagine, ordinate, referenziate dalla più vecchia -> alla più recente
  - Quindi in caso si abbia 7,0,1 vuol dire che la più vecchia è 7 che va rimossa.
- Data la sequenza sopra (7,0,1) e viene referenziata la pagina 0, la sequenza in coda diventa 7,1,0 perchè viene aumentato il contatore per quella pagina
- La LRU genera 12 page fault.
- FIFO generava 15 page fault.
- **LRU risulta migliore** rispetto all'algoritmo FIFO perchè si avvicina di più alla soluzione ottimale OPT

Anomalia FIFO di Belady

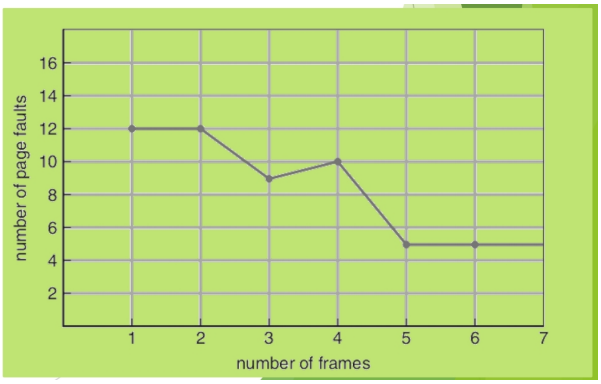
Nell'algoritmo FIFO si erano generati 15 fault. Si diceva che più sono i frame, meno sono i fault.

Data la sequeza *1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5* e si va a valutare il numero di fault **usando 3 frame e usando 4 frame** (ci si aspetta che il numnero di fault con 3 frame sia maggiore del numero di fault con 4 frame):

- Con **3 frame** si generano **9 page fault**
- Con **4 frame** si generano, invece, **10 fault**

Si presenta l'***ANOMALIA DI BELADY*** cioè **aumentando** i frame disponibili, **aumentano** i page di fault
L'algoritmo FIFO (e i suoi derivati) presenta questa anomalia

Si ha, appunto, il seguente grafico:



L'algoritmo LRU NON soffre dell'anomalia di Belady e gode della ***PROPRIETA' DI INCLUSIONE***: cioè l'insieme delle pagine caricate avendo n frame è incluso in quello che si avrebbe avendo n+1 frame. Cioè più aumento i frame e meno page fault si avranno

In generale...

- **NFU** e **Aging** non soffrono dell'anomalia di Belady ma godono della proprietà di inclusione
- **Seconda Chance** e **Clock** (derivati da FIFO) soffrono dell'anomalia di Belady (più frame, più fault) (*si riduce a FIFO*)
- NRU soffre dell'anomalia di Beldy e si riduce a FIFO

Riepilogo Algoritmi di Sostituzione

Solo l'AGING tiene in considerazione l'ATTIVITA' PASSATA di ogni singola pagina

Allocazione dei Frame

*Appena ci si riferisce ad una **pagina per la prima volta**, all'inizio del programma, si avrà un page fault.* Ogni pagina avrà delle variabili locali e riferimenti ad altre pagine, quindi aumentano i page fault collegati alla pagine iniziale. Il numero di page fault, quindi, inizialmente sarà alto (*perchè inizialmente non si hanno tutte le informazioni necessarie*) e poi andranno a ridursi.

- La *strategia* di ***ASSEGNARE PAGINE SU RICHIESTA*** prende il nome di ***DEMAND PAGING***.
- Il numero di frame assegnati è determinante per l'esecuzione del processo e per le prestazioni del sistema (*meno frame si hanno, più fault si generano e viceversa ->* tranne *FIFO*)

Quanta memoria bisogna allocare per i processi?

- Una piccola porzione serve per il sistema operativa
- Il resto serve per l'esecuzione delle varie attività
- Con il *Demand Paging*, il primo frame genera fault. Si avranno fault **fino a quanto tutto sarà caricato in memoria**
  - Per questo motivo si avranno **SICURAMENTE** `n` fault, con `n` = dimensione della frame.

IDEALMENTE l'idea è quella di avere dei frame da gestire i frame:

- **IN PIENO**, quindi fault fino al riempimento della frame
- **RISERVARE DEI FRAME LIBERI**. Quando si ha page-fault, si avrà già un frame libero da assegnare. Vi sarà, quindi, un **processo demone** che terrà questi frame liberi)

> Questo ragionamento sottintende che c'è ***UN SOLO PROCESSO***

Strategie di allocazione

Vi sono 2 strategie e, qualsiasi essa si utilizzi, devono sottostare ad alcuni limiti e condizioni:

- Ad ogni processo **si deve assegnare** un **NUMERO MINIMO DI FRAME**
- Ad ogni processo si deve assegnare un **NUMERO MASSIMO DI FRAME** al quale esso può accedere
  Come si assegnano questi frame? si tengono in conto i seguenti ***VINCOLI***:
- Non si può allocare *un numero superiore* al numero totale di frame
- ad ogni processo spetta *un numero minimo di frame* che servono per lo svolgimento delle operazioni

Strategia: allocazione equa

Il numero di frame assegnati dipende dal numero di fault che ne vengono fuori.

- Assegna **un numero di frame UGUALE** per ciascun processo
- Tutti i processi non hanno le stesse dimensioni fra loro.
- Quindi **questa strategia non è del tutto corretta** perchè potrebbe capitare che un processo **A** sia piccolo (quindi può avere frame maggiori del necessario) e un processo **B** che hanno un numero di frame assegnati minori rispetto a quelli necessari ad esso
- L'idea di base di questa strategia è corretta solo ***IDEALMENTE***

Per questo motivo si allocano i frame in modo ***PROPORZIONALE***

Strategia: allocazione proporzionale

Vengono allocati i frame in modo da RIDURRE IL NUMERO DI FAULT complessivamente (anche aumentando di poco il numero di fault di un singolo processo)
Si tiene in considerazione la seguente procedura:

- S = somma di tutte le dimensioni di tutti i processi
- stima del numero di frame da assegnare al processi a_i
- $a_i = (s_i/S)\times m$

Si può osservare che l'allocazione proporzionale è più idonea e l'allocazione (in generale) tiene in considerazione i processi come se fossero **TUTTI UGUALI**. I processi non sono tutti uguali perchè possono esistere processi con priorità diversa rispetto ad altri.

Allora, in questo caso, si usa un'allocazione proporzionale **RISPETTO ALLE PRIORITA'** oppure una combinazione di allocazione (*giustamente pesata*) fra **DIMENSIONE E PRIORITA'**

- Dal *numero di frame assegnati* ad un processo **dipende l'efficienza dell'intero sistema**. Il numero di page fault che un processo produce determina l'efficienza del sistema. (ogni page fault causa un rallentamento delle operazioni)

> *Quando si fa una sostituzione di una pagina, la pagina sostituita come si sceglie?*
> Ogni processo ha delle pagine assegnate su cui lavora

Gli algoritmi di sostituzione di pagine possono agire secondo:

- allocazione locale: la pagina da rimuovere deve essere locale a quel determinato processo
- allocazione globale: la pagina da rimuovere deve essere fra tutte le possibili pagine

Esempio:

	Age
A0	10
A1	7
A2	5
A3	4
A4	6
A5	3
B0	9
B1	4
B2	6
B3	2
B4	5
B5	6
B6	12
C1	3
C2	5
C3	6

(a)

Si debba sostituire una pagina, considerando Age = ultimo referenziamento:

- Quale pagina si sostituisce? Le pagine fra il processo A? (processo con Age più basso)
- Se tolgo A5 allora si parla di **sostituzione locale** (rimozione della pagina fra le pagine assegnate a quel processo)
- Se tolgo B3 allora si parla di **sostituzione globale** (rimozione della pagina fra tutte le pagine disponibili -> **anche di altri processi**)

In caso di allocazione globale, il numero di frame allocati non è più fisso ma diventa ***DINAMICO*** (in caso di page fault -> capiterà che **un processo avrà un frame IN PIU'** e **un processo avrà un frame IN MENO**)

In caso di allocazione locale, i frame allocati a quel processo sono ***STATICI*** (quindi non cambiano mai)

In caso di allocazione globale, il **numero di fault** che ne vengono fuori non dipende più dal singolo processo ma ***da fattori non controllabili da tale algoritmo*** (perchè aumenta la possibilità di fault per i processi che perdono frame). Diversamente dall'allocazione locale, il numero di frame dipende esclusivamente dal **processo stesso**, quindi **controllabili da lui**

> Complessivamente l'allocazione globale ***sembra essere la scelta più efficiente*** a livello di produttività del sistema. Per questo motivo è il metodo più comune utilizzato

L'obiettivo, per limitare i page fault, è quello di cercare costantemente di ***assegnare il numero minimale necessario di frame*** per avere una maggiore produttività.

Ogni singolo processo non deve scendere sotto una ***QUOTA MINIMA DI FRAME ASSEGNATI***. Se ciò accade, il processo viene **SOSPESO** e attende l'assegnazione dei frame minimi

- Se si scende sotto il numero minimo di frame necessari per un processo, ***vi sono frequenti e consecutivi page-fault***. Si parla allora di ***TRASHING***.

Quando il trashing **riguarda tutti i processi**, si parla di ***SISTEMA IN SOVRACCARICO***

> *Se ho solo una pagina e quindi un frame, allora i page-fault sono per ogni richiesta di pagina perchè la stessa locazione deve essere sostituita*

- Se i fault iniziano a diventare **troppi** (trashing) vuol dire che **si devono allocare più frame** perchè sono meno di quelli minimali per quel processo
- Se i fault iniziano a diventare **troppo pochi**, allora vuol dire che quel processso ha **troppi frame assegnati**
- Si deve mantenere un **numero di fault** in un ***determinato RANGE*** che consentono prestazioni migliori per l'intero sistema