

Gestiunea memoriei-Eliberarea

Presupunem Liber – sortata crescator dupa adresele blocurilor din lista.

Fie b – blocul de eliberat.

Fie pred, succ blocuri din lista Liber a.i. $\text{adr}(\text{pred}) < \text{adr}(b) < \text{adr}(\text{succ})$

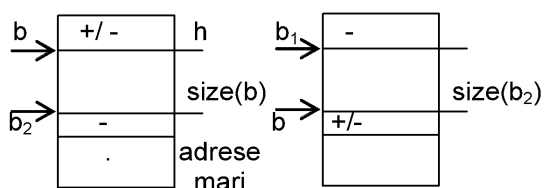
Daca : $\text{adr}(\text{succ}) = \text{adr}(b) + \text{lg}(\text{header}) + \text{size}(b)$

atunci putem fuziona cele doua blocuri \Rightarrow fuzionare in amonte

Daca : $\text{adr}(\text{pred}) = \text{adr}(b) - \text{size}(\text{pred}) - h \Rightarrow$ fuzionare in aval

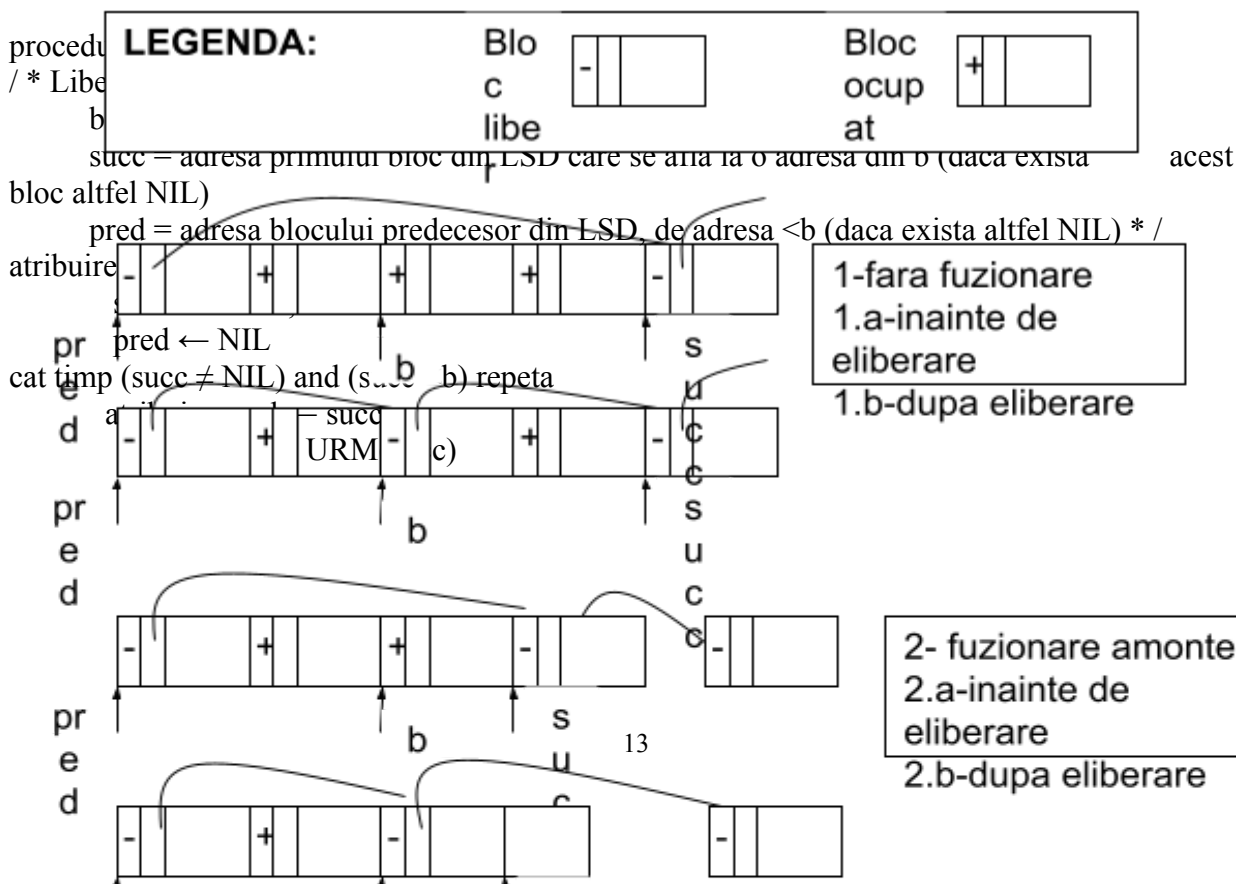
Daca se verifica ambele conditi va avea loc o fuzionare dubla.

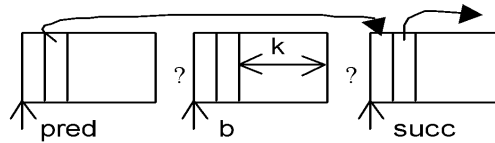
Mai jos: $b_1 = \text{pred}$ si $b_2 = \text{succ}$



Cateva situatii posibile sunt prezentate in diagramele de mai jos :

- in figura 1 este cazul de eliberare a blocului b dar nu se poate face fuziune
- in fig. 2 se poate face fuziune in amonte
- in fig. 3 se prezinta cazul de fuziune dubla

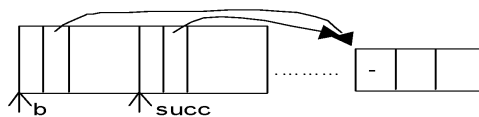




/* in acest moment avem :
 \Rightarrow Testez conditiile de fuziune */

daca $\text{succ} \neq \text{NIL}$ /* i.e. (\exists) in LSD un bloc la o adresa $> b$ */
 atunci daca $(b+h+k = \text{succ})$ /* conditia fuziune “amonte” */
 atunci $\text{size}(b) \leftarrow k + \text{size}(\text{succ}) + h$;
 $\text{urm}(b) \leftarrow \text{urm}(\text{succ})$;

i.e. /*

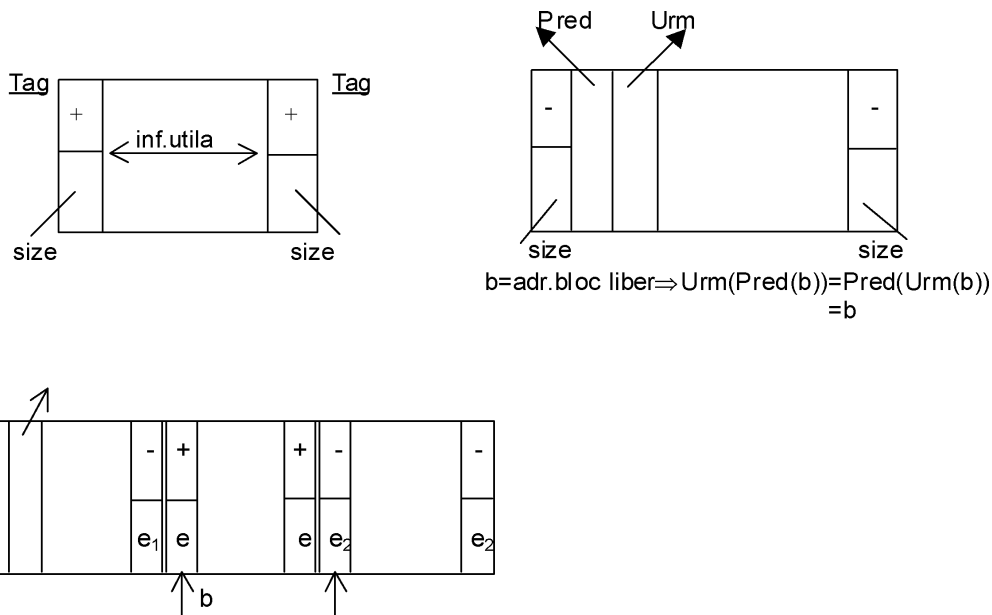


/*
 altfel $\text{urm}(b) \leftarrow \text{succ}$
 altfel $\text{urm}(b) \leftarrow \text{NIL}$ /* nu exista in LSD alt bloc dupa b */
 daca $\text{pred} \neq \text{NIL}$ /* i.e. (\exists) bloc in LSD la o adresa $< b$ */
 atunci daca $\text{pred} = b - \text{size}(\text{pred}) \leftarrow \text{size}(\text{pred}) - h$ /*conduce la fuziune in “aval” */
 atunci $\text{size}(\text{pred}) \leftarrow \text{size}(\text{pred}) + \text{size}(b) + h$;
 $\text{urm}(\text{pred}) \leftarrow \text{urm}(b)$;
 altfel $\text{urm}(\text{pred}) \leftarrow b$;
 altfel $\text{Liber} \leftarrow b$;
 sfarsit.

OBS : \rightarrow La alocare preferam ca lista LSD sa fie pastrata sortata crescator in functie de lungimea blocurilor din lista .

\rightarrow La eliberare in schimb, daca blocurile din lista apar in ordinea adreselor lor \Rightarrow testez mai usor adresele de fuziune.

Daca aleg alta structura pentru blocuri , pot sa pastrez lista sortata dupa lungime si pot sa testez comod si conditia de fuziune.



Testez conditia de fuziune :

a) amonte $\text{Tag}(b + 2h + \text{size}(b)) = '-'$

aval $\text{Tag}(b-1) = '-'$

1

OBS : - Ca urmare a fuzionarii este posibil sa fie necesara sortarea listei LSD .

→ La oricare din metodele de pana acum apare un dezavantaj : fragmentarea spatiului de memorie.

Buddy System (Sistemul cu “camarazi” (Knuth))

-

Principiul metodei :

→ Cu aceasta metoda se alocă blocuri de memorie a caror **lungime trebuie sa fie o putere a lui 2** (i.e. $2, 2', 2^2, \dots$)

Daca se afce o cerere de alocare cu o lungime $\neq 2^k \Rightarrow$ se alocă primul bloc acoperitor ca lungime si care are o lungime = cu o putere a lui 2.

→ Blocul de memorie din care se fac alocările, este presupus de dimensiune 2^M .

→ Evidenta blocurilor libere, se tine cu ajutorul unui vector de liste.

→ Fiecare componenta i a acestui vector mentine lista a blocurilor libere de dimensiune 2^i sau lista vida daca in momentul considerat nu exista nici un bloc liber de dimensiune 2^i . → Initial (\exists) un singur bloc, avand dimensiunea intregii memorii disponibile 2^M .

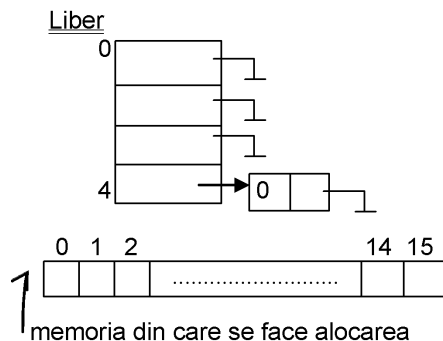
→ La o cerere de alocare a unui bloc de dimensiune 2^k , se caută un bloc liber de aceasta dimensiune.

daca exista \Rightarrow se alocă acest bloc

daca nu exista, dar (\exists) un bloc de dimensiune mai mare, se incepe un proces de injumatatire, plecand de la acest bloc, proces care se continua pana se obtine un bloc de dimensiune 2^k .

OBS : Cand un bloc este divizat in doua , cele doua jumatatii poarta numele de “camarazi” (“muguri”).

EX : $M = 4$



OBS : \rightarrow O intrare i din vectorul Liber (i.e. Liber [i]), este un pointer spre lista blocurilor libere de dimensiune 2^i .

\rightarrow Fiecare element al listei contine in primul camp adresa blocului (adresa de inceput).

OBS : \rightarrow Cand un bloc de dimensiune 2^{k+1} se injumatateste rezulta doua blocuri (“camarazi”, “muguri”) pe care le notam $B_k(A)$ si $B_k(A')$ care au proprietatile :

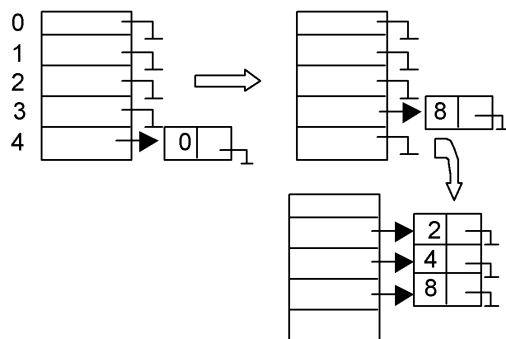
fiecare bloc are dimensiune $= 2^k$

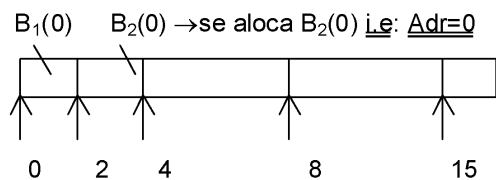
$A \bmod 2^k = 0$ si $A' \bmod 2^k = 0$ (i.e. adresa la care incepe oricare din cele doua blocuri (A sau A') este multiplu de 2^k).

OBS : $\left\{ \begin{array}{l} K = \text{ordinul blocului} \\ A = \text{adresa blocului} \end{array} \right\} \Rightarrow B_k(A) \rightarrow \text{bloc de ordinul } K \text{ si adresa } A.$

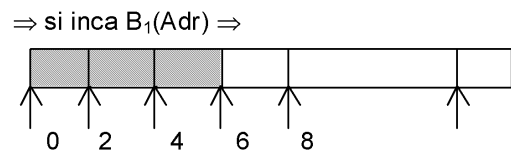
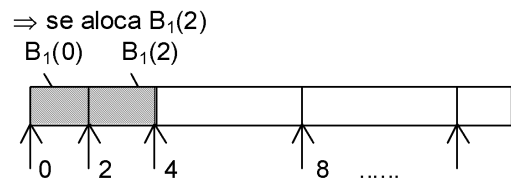
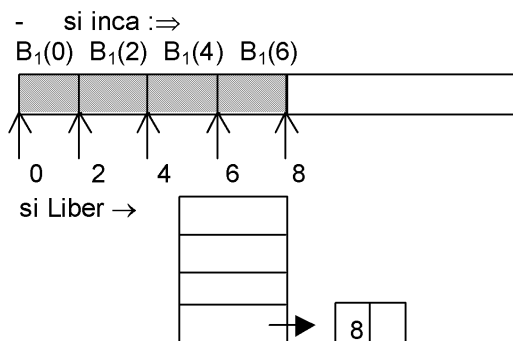
Fie in exemplul nostru o cerere de alocare a unui bloc de dimensiune $= 2$ cuvinte ; i.e. cerere de determinare $B_1(Adr)$ si vreau sa aflu adresa Adr la care se face alocarea: \Rightarrow avem succesiv:

!!!!!!



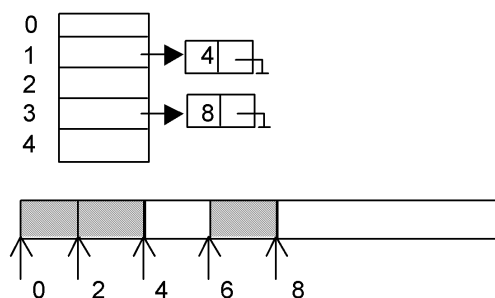


OBS : Injumatatirea se continua cu “camaradul”
 din stanga pana la 2^k , iar cel din dreapta e introdus ca bloc liber.
 Acum presupun o noua cerere $B_1(Adr)$:

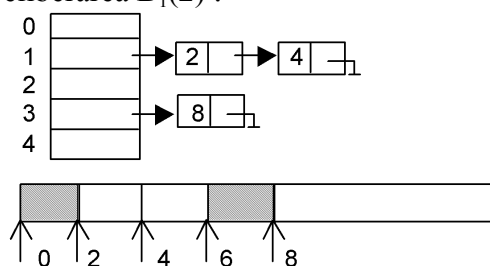


Eliberarea

eliberarea blocului $B_1(4)$:



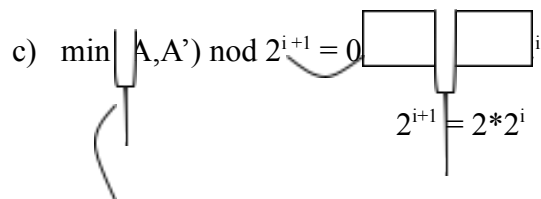
eliberarea $B_1(2)$:



OBS : \rightarrow pentru a respecta algoritmul nu pot sa fuzioneze doua blocuri vecine, decat daca ele sunt “camarazi” i.e. : au provenit din divizarea aceluasi bloc (la noi , $B_1(2)$ si $B_1(4)$ nu sunt “camarazi” intre ei).

DEF : Fie $B_i(A)$ si $B_j(A')$ doua blocuri eliberate. Conditile ca cele doua blocuri sa poata fuziona : sa aiba aceeasi dimensiune $\Rightarrow i = j$

$|A - A'| = 2^i$ (blocuri adiacente de dimensiune 2^i)



adresa blocului care ar rezulta dupa fuziune (!!sa fie la multiplu 2^{i+1})
 \rightarrow La noi in ultimul caz : $\min(A, A') = 2$; $2 \bmod 4 \neq 0$!!

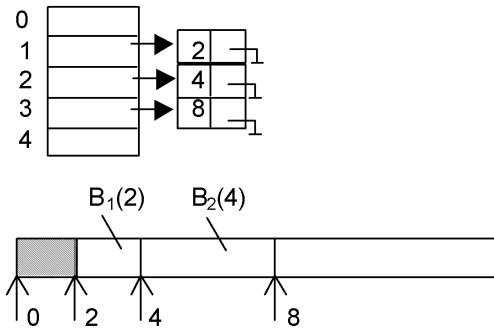
· Si acum o noua eliberare : $B_1(6)$.

Se observa ca $B_1(6)$ si $B_1(4)$ indeplinesc conditiile de fuziune:

aceiasi dimensiune

$$|6 - 4| = 2 = 2^1$$

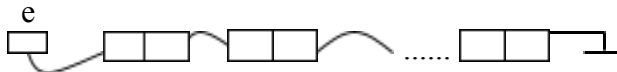
$$\min(A, A') = \min(4, 6) = 4 \text{ si } 4 \bmod 2^{1+1} = 0$$



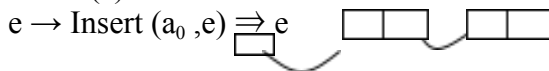
Pentru evidenta spatiului disponibil ,am o variabila TabLis : array $[0..M]$ of lista (tablou de lista) unde $2^M =$ dimensiunea totala a spatiului din care fac alocarile.

Presupun ca pentru Lista , am definiti urmatoorii operatori: First , Insert , Delete.

Ex:



atunci : First (e) $\rightarrow a_1$



$e \rightarrow$ Delete (a_i, e) \Rightarrow elimina a_i din e.

Presupun ca se fac cereri de alocare $B_k(A)$ i.e. se cere alocarea unui bloc de dimensiune = 2^k .Daca pot sa fac alocarea intorc in A adresa blocului alocat, modificand corespunzator TabLis.Daca nu se poate face alocarea ,intorc NIL.

procedura Alocare (TabLis , M,k ; TabLis,Adr) este $j \leftarrow k$;

cat timp ($j \leq M$) and ($\text{TabLis}[j] = \text{NIL}$) executa

$j \leftarrow j + 1$;

daca $j > M$

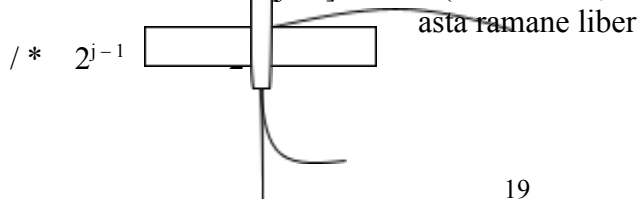
atunci $\text{ADR} \leftarrow \text{NIL}$ /* nu mai exista spatiu */

altfel $\text{ADR} \leftarrow \text{First}(\text{TabLis}[j])$; /* ADR aici fac alocare */

$\text{TabLis}[j] \leftarrow \text{Delete}(\text{ADR}, \text{TabLis}[j])$;

cat timp($j < k$) executa

$\text{TabLis}[j-1] \leftarrow \text{Insert}(\text{ADR} + 2^{j-1}, \text{TabLis}[j-1])$



$$\uparrow \text{ADR} \quad \text{ADR} + 2^{j-1} \\ 2^j \text{ cu } j > k$$

*/

$$j \leftarrow j - 1 ;$$

sfarsit.


OBS : First $\begin{Bmatrix} \text{Car} \\ \text{Top} \end{Bmatrix}$ Insert $\begin{Bmatrix} \text{Push} \\ \text{Cons} \end{Bmatrix}$ Delete $\begin{Bmatrix} \text{Cdr} \\ \text{Pop} \end{Bmatrix}$

procedura Eliberare (TabLis,M,k,A) este

$j \leftarrow k;$

$\text{gata} \leftarrow \text{Fals};$





cat timp ($j \leq M$) and (not gata) executa

daca  exista $A1 \in \text{TabLis}[j]$ care poate fuziona cu A i.e.: $|A - A1| = 2^j$
and $\min(A, A1) \bmod 2^{j+1} \equiv 0$

atunci /* $A1 \neq \text{NIL}$ */

$\text{TabLis}[j] \leftarrow \text{Delete}(a1, \text{TabLis}[j])$

$A \leftarrow \min(A, A1);$

/*   sau   */

$j \leftarrow j - 1 ;$ /* \Rightarrow un bloc liber de dimensiune dubla care poate mai
fuzioneaza si el */

altfel $\text{TabLis}[j] \leftarrow \text{Insert}(A, \text{TabLis}[j])$

$\text{gata} \leftarrow \text{Adevarat};$

sfarsit.