Universitatea Politehnica din București
Facultatea de Automatică și Calculatoare
Departamentul de Calculatoare

ARBORI DE REGĂSIRE

Introducere

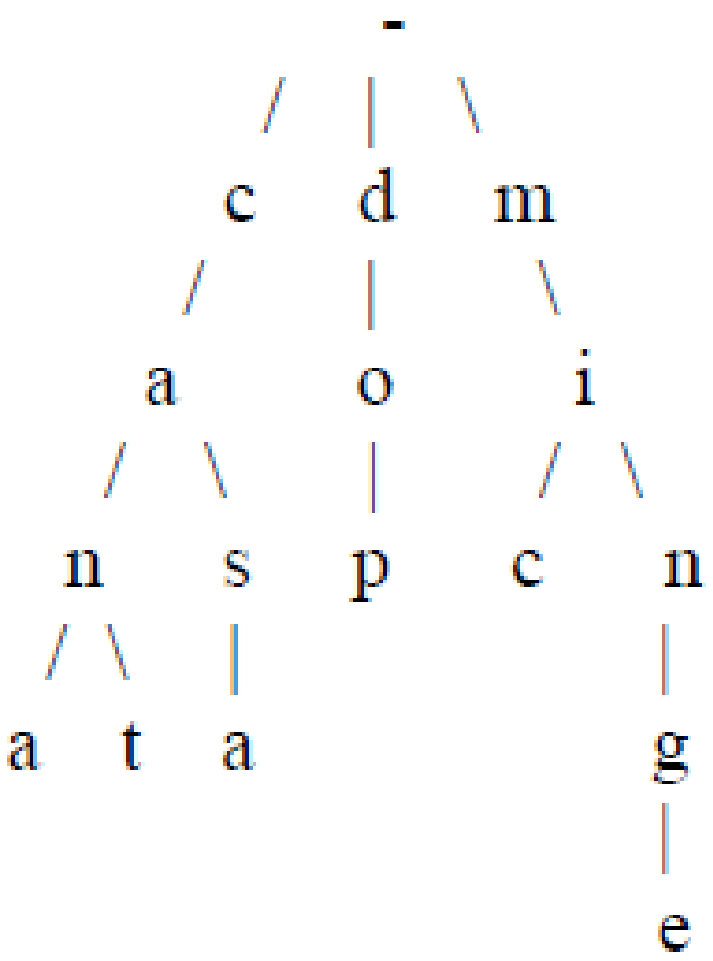
- **Arborii de regăsire** sunt arbori **multicăi**
- Acești arbori sunt denumiți și **trie**, denumire provenită din termenul re**trie**val (regăsire)

Utilizare

- Un **arbore de regăsire** este un arbore folosit pentru memorarea unor șiruri de caractere sau unor șiruri de biți de lungimi diferite, dar care au în comun unele subșiruri, ca prefixe

Exemplu

- Arbore de regăsire construit cu șirurile: cana, cant, casa, dop, mic, minge



Observații

- Nodurile unui arbore de regăsire pot conține sau nu date
- Un **șir** este o cale de la rădăcină la un nod frunză sau la un nod interior
- Pentru șiruri de biți arborele de regăsire este **binar**
- Pentru șiruri de caractere arborele de regăsire **nu este binar** (numărul de succesori ai unui nod este egal cu numărul de caractere distincte din șirurile memorate)

Observații

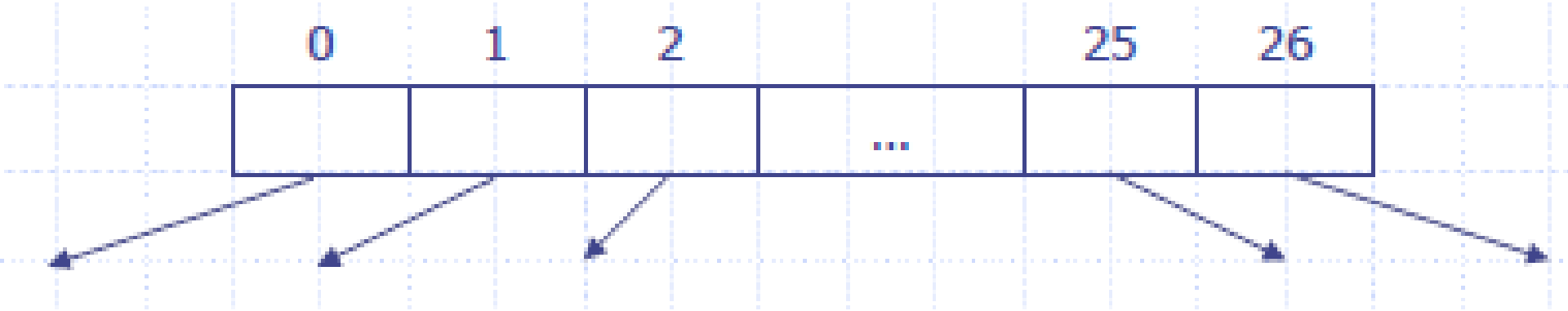
- Considerăm arbori de regăsire utilizați pentru memorarea de cuvinte alcătuite din litere mari în intervalul ‘A’ ... ‘Z’
- Alegerea intervalului de litere este importantă în implementarea unui arbore de regăsire, deoarece de acest interval depinde structura nodurilor arborelui de regăsire

Structura unui nod

- Fiecare nod are un **număr de subarbori** egal cu **lungimea intervalului de caractere** (‘A’ ... ‘Z’) plus o unitate
- Deoarece între ‘A’ și ‘Z’ sunt **26** de caractere (inclusiv ‘A’ și ‘Z’, folosind alfabetul limbii engleze), rezultă că fiecare nod al arborelui de regăsire va avea **27** de **pointeri spre fii** (dispuși, de exemplu, sub forma unui tablou)

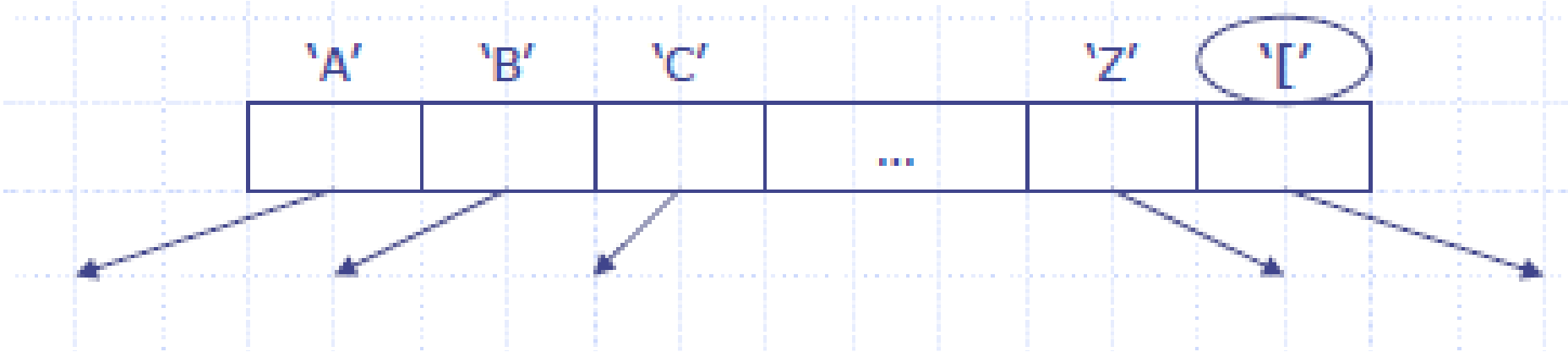
Reprezentarea unui nod

- Un nod al arborelui de regăsire este:



Reprezentarea unui nod

- Deoarece fiecare nod al arborelui de regăsire este legat de intervalul de caractere ‘A’ ... ‘Z’ se reprezintă nodul prin:



Observații

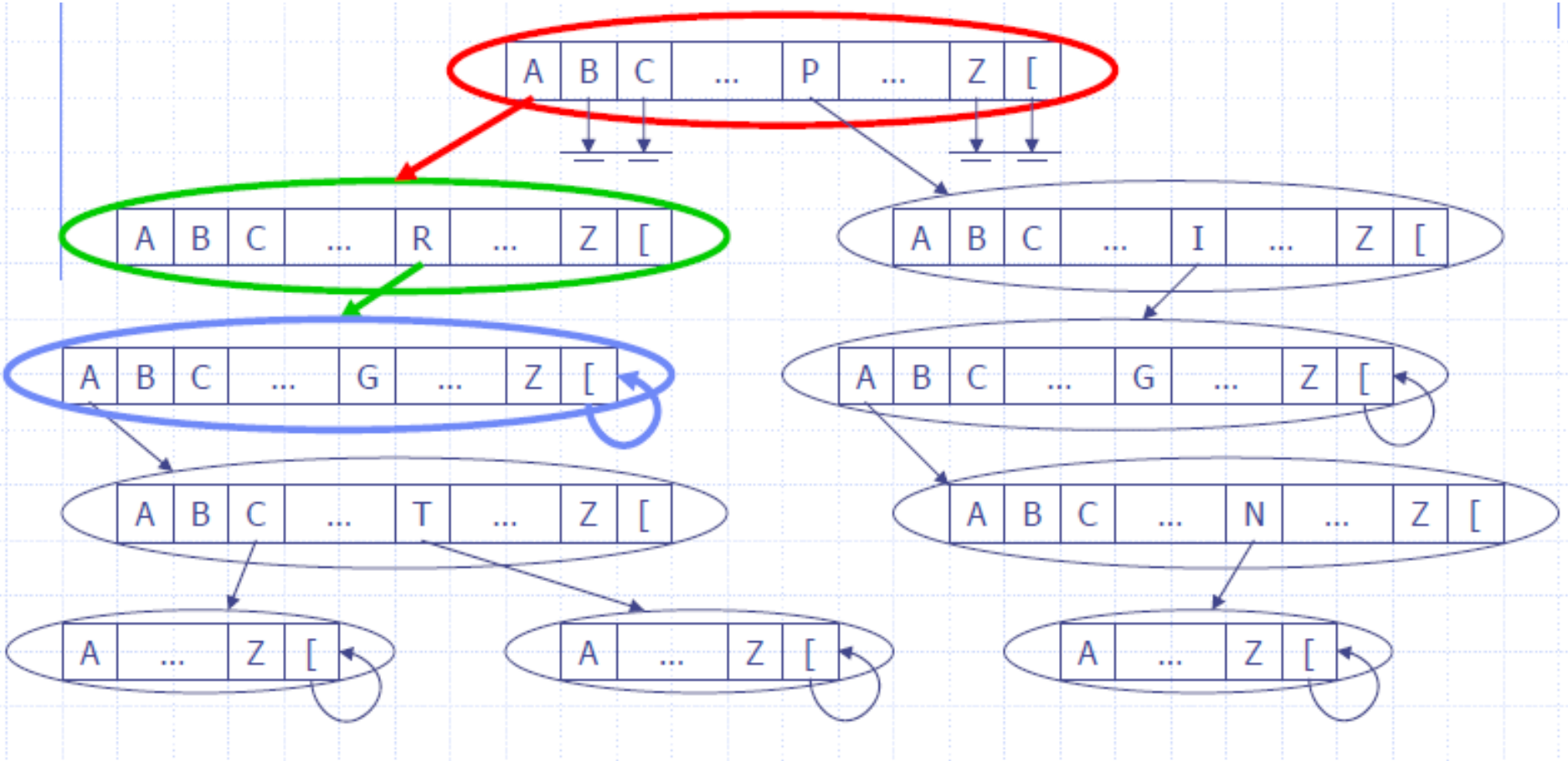
- Fiecare pointer din nod corespunde unui caracter din intervalul ‘A’ ... ‘Z’ cu excepția ultimului pointer
- Fiecare nod conține cu o unitate mai mulți pointeri decât caractere in intervalul ‘A’ ... ‘Z’
- Ultimul pointer din nod corespunde caracterului ‘[’

Observații

- Caracterul ‘[’ reprezintă caracterul care urmează lui ‘Z’ în codificarea ASCII
- Se extinde intervalul ‘A’ ... ‘Z’ la dreapta cu un caracter, obținând intervalul ‘A’ ... ‘[’ (acesta este intervalul efectiv)
- Acest ultim caracter este important și se folosește drept caracter **terminator**

Exemplu

- Arbore de regăsire care conține cuvintele: AR, ARAC, ARAT, PI și PIAN



Observații

- Chiar dacă fiecare nod conține un tablou de 27 de caractere (de la ‘A’ până la ‘[’), de fapt, avem un tablou de 27 de pointeri
- Caracterele sunt reprezentate doar pentru a simplifica înțelegerea structurii
- Toți pointerii care nu sunt reprezentați prin săgeți se consideră că sunt nuli
- Doar pointerii nuli ai rădăcinii sunt reprezentați explicit prin săgeți

Observații

- Pentru a înțelege această structură, trebuie să stabilim criteriul de apartenență al unui cuvânt la mulțime
- Pentru a studia apartenența cuvântului AR, se consideră șirul “AR[” (se completează cuvântul cu caracterul ‘[’)

Observații

- Se pornește de la rădăcină
  - Nodul curent este nodul **roșu**
  - Pointerul corespunzator caracterului ‘A’ (pointerul **roșu**) trebuie să fie nenul
  - Se urmează acest pointer și se ajunge într-un nod nou (nodul **verde**)
  - Pointerul corespunzător caracterului ‘R’ (pointerul **verde**) trebuie să fie nenul
  - Se urmează acest pointer și se ajunge într-un nod nou (nodul **albastru**)
  - Pointerul corespunzător caracterului ‘[’ (pointerul **albastru**) trebuie să fie nenul

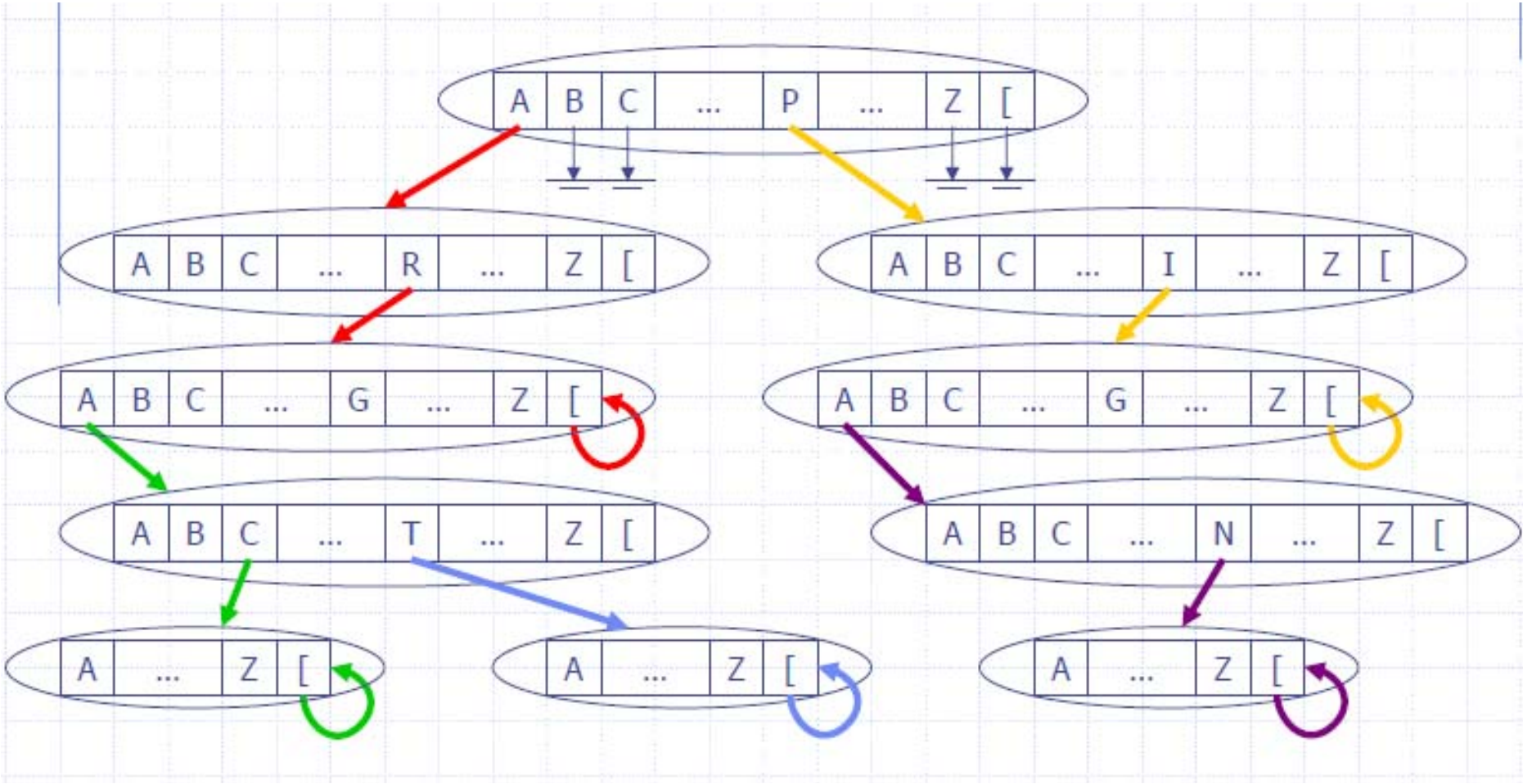
Observații

- Cum toți pointerii pe calea de căutare sunt nenuli, concluzia este: cuvantul “AR” aparține mulțimii
- Dacă cel puțin un pointer pe calea de căutare ar fi fost nul, cuvântul “AR” nu ar fi aparținut mulțimii
- Când vorbim de toți pointerii, ne referim la cuvântul extins “AR[” (completat cu caracterul terminator)

Observații

- Pointerul corespunzător caracterului ‘[’ (atunci când este nenul) este întotdeauna un pointer în buclă (un pointer chiar la nodul din care face parte pointerul respectiv)
- Nu contează nodul spre care pointează acel pointer, ideea este că pointerul trebuie să pointeze spre un nod valid (să nu fie nul)

Căutare în arbore de regăsire



Căutare în arbore de regăsire

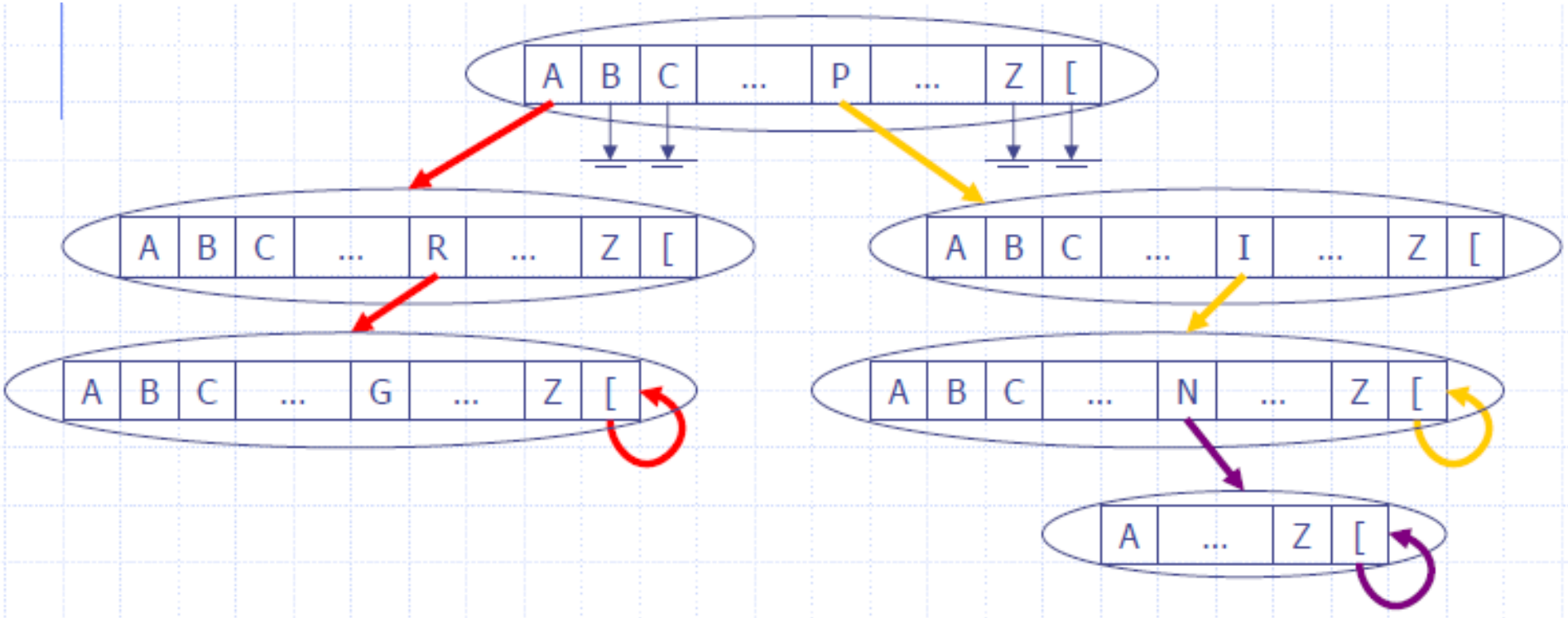
- Secvențele de pointeri corespunzătoare celor 5 cuvinte sunt (pornind de la rădăcină):
  - Cuvântul “AR”: **ROȘU** – **ROȘU** – **ROȘU**
  - Cuvântul “ARAC”: **ROȘU** – **ROȘU** – **VERDE** – **VERDE** – **VERDE**
  - Cuvântul “ARAT”: **ROȘU** – **ROȘU** – **VERDE** – **ALBASTRU** – **ALBASTRU**
  - Cuvântul “PI”: **PORTOCALIU** – **PORTOCALIU** – **PORTOCALIU**
  - Cuvântul “PIAN”: **PORTOCALIU** – **PORTOCALIU** – **MOV** – **MOV** – **MOV**

Observații

- Există 5 căi de căutare în arbore, corespunzătoare celor 5 cuvinte
- Pentru inserarea sau ștergerea de cuvinte din mulțime, trebuie create respectiv eliminate secvențele de pointeri corespunzătoare

Inserare în arbore de regăsire

- Pentru inserarea unui cuvânt, se consideră că avem un arbore de regăsire conținând doar cuvintele AR (pointeri **roșii**) și PI (pointeri **portocalii**) și dorim inserarea cuvântului PIN


A B C ... P ... Z [
A B C ... R ... Z [
A B C ... I ... Z [
A B C ... G ... Z [
A B C ... N ... Z [
A ... Z [

Observații

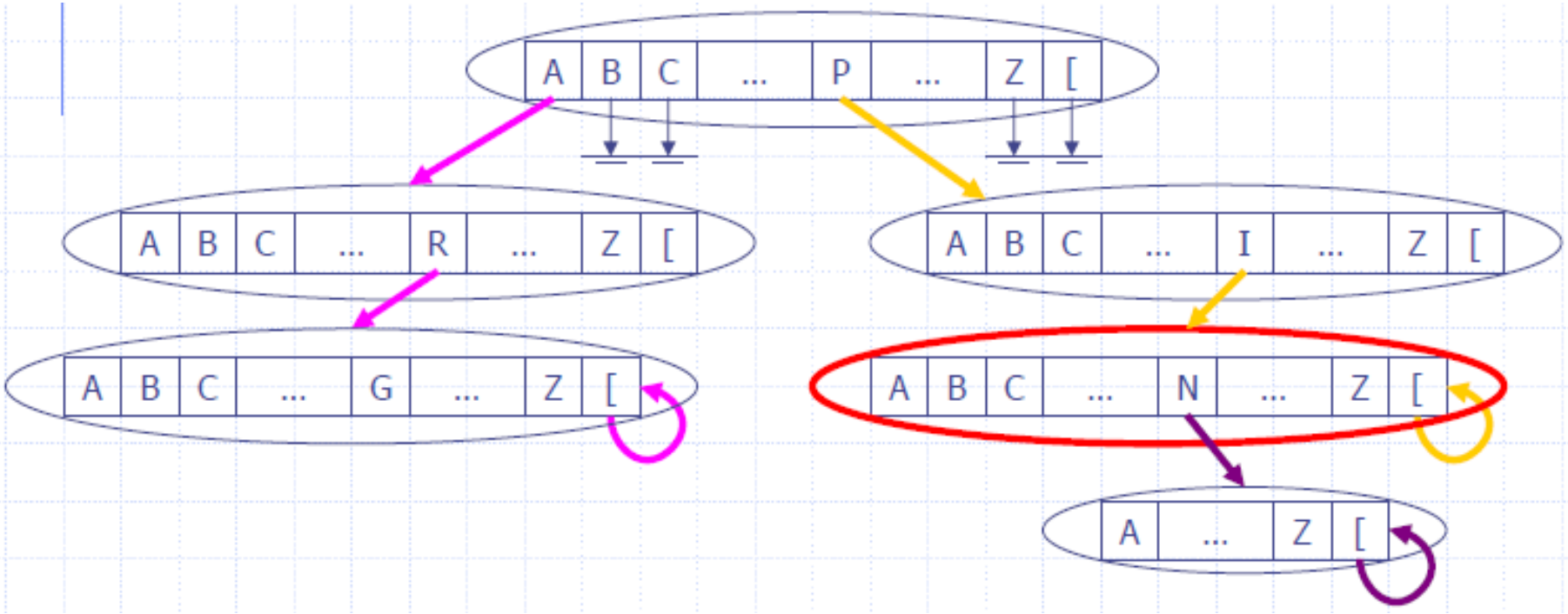
- Se pornește de la rădăcină și se parcurge calea indicată de pointerii corespunzători caracterelor ‘P’, ‘I’, ‘N’ și ‘[’
- Dacă cuvântul “PIN” nu face parte din mulțime, înseamnă că cel puțin unul dintre cei 4 pointeri amintiți sunt nuli
- Procesul de inserare trebuie să creeze nodurile necesare pe calea de căutare, astfel încât cei 4 pointeri să devină toți nenuli

Observații

- Pointerii corespunzători caracterelor ‘P’ și ‘I’ sunt deja nenuli, datorită cuvântului “PI”, deci nu trebuie făcut nimic în plus pentru acești pointeri
- Se creează un nod nou, corespunzător caracterului N, iar în acel nod nou, pointerul corespunzător caracterului terminator ‘[’ trebuie să nu fie nul, deci se creează bucla corespunzătoare

Ștergere din arbore de regăsire

- Ștergerea unui cuvânt din mulțime necesită 2 cazuri, după cum este necesar sau nu să eliberăm memorie
- Vrem să ștergem cuvântul “PI” din arbore


A B C ... P ... Z [
A B C ... R ... Z [
A B C ... G ... Z [
A B C ... I ... Z [
A B C ... N ... Z [
A ... Z [

Observații

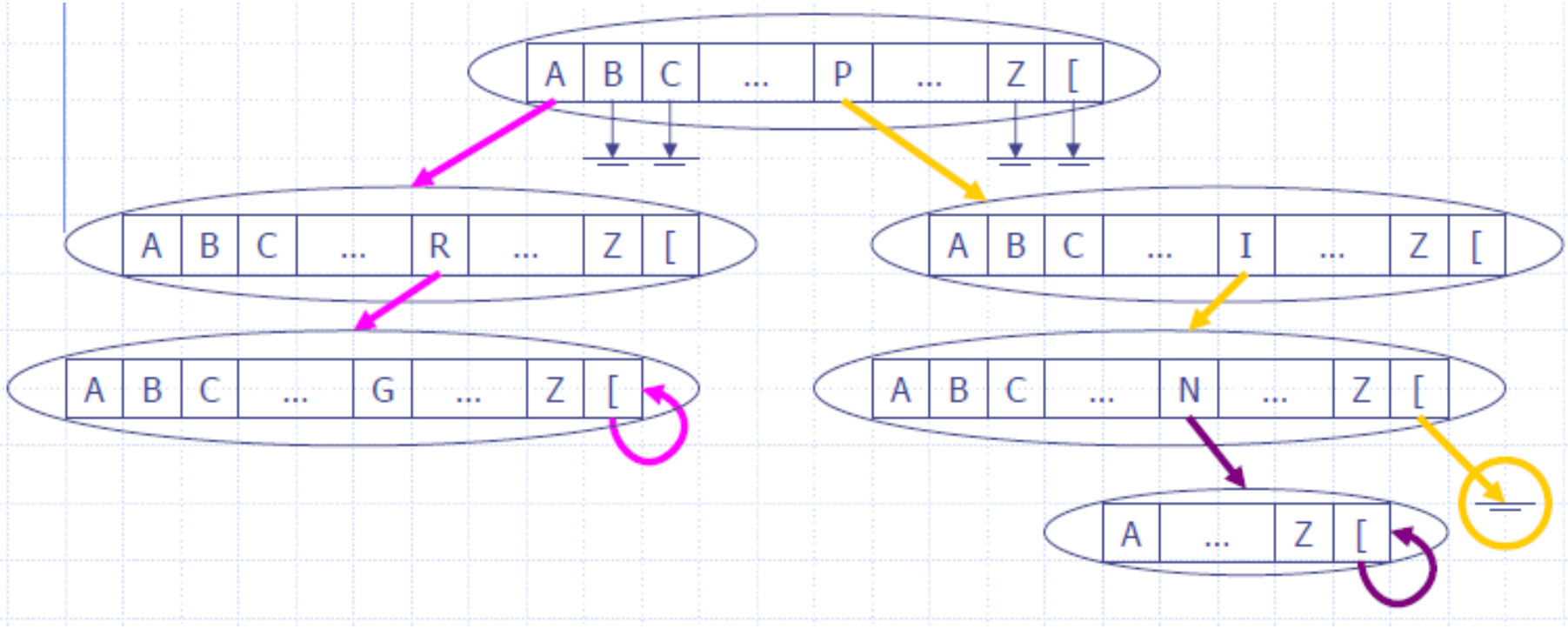
- Se pornește de la rădăcină, se verifică dacă mulțimea conține cuvântul pe care vrem să-l ștergem
- Se parcurg pointerii corespunzători caracterelor ‘P’, ‘I’ și ‘[’ (adică drumul **portocaliu**) și se ajunge la concluzia că cuvântul “PI” aparține mulțimii
- Pentru a șterge cuvantul “PI” trebuie să anulăm pointerul corespunzător caracterului ‘[’ (bucla **portocalie**)

Observații

- Nodul curent este nodul **roșu**
- Cum nodul curent mai conține pointeri nenuli, înseamnă că mai există cuvinte care încep cu prefixul “PI”, deci ne oprim aici
- Practic, tot ce a trebuit să facem a fost să parcurgem calea de căutare și să anulăm ultimul pointer

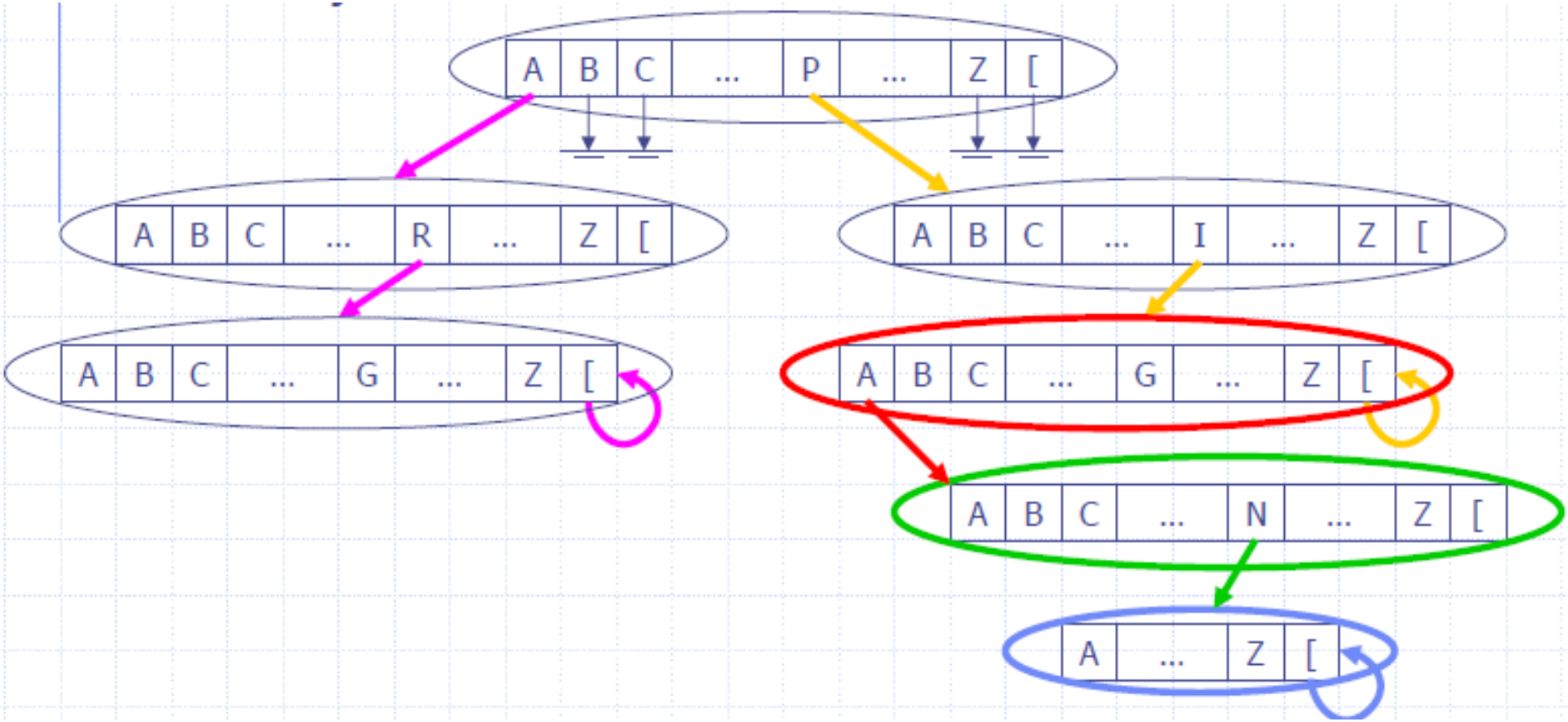
Ștergere din arbore de regăsire

- Arborele obținut este (modificările sunt încercuite):



Ștergere din arbore de regăsire

- Vrem să ștergem cuvântul “PIAN” din arbore:



Observații

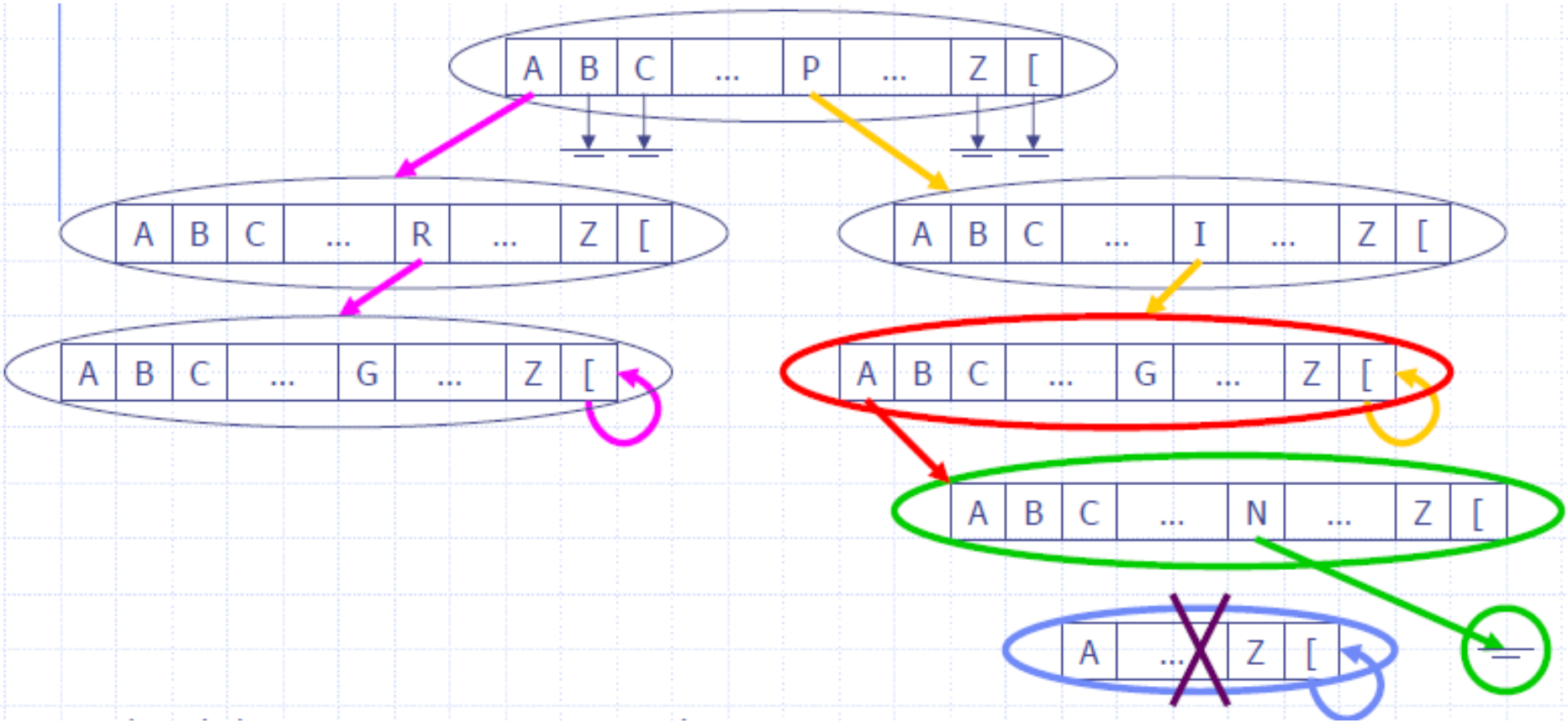
- Urmând pointerii corespunzători caracterelor ‘P’, ‘I’, ‘A’, ‘N’ și ‘[’, ajungem în nodul **albastru** și concluzionăm că cuvântul “PIAN” aparține mulțimii
- Anulăm pointerul corespunzător caracterului ‘[’ din nodul **albastru**

Observații

- Deoarece nodul curent (nodul **albastru**) nu mai conține pointeri nenuli (toți sunt nuli), înseamnă că nu mai există cuvinte care încep cu prefixul “PIAN”, deci nu are rost să menținem spațiul ocupat pentru nodul **albastru**

Observații

- Urcăm în nodul părinte (nodul **verde**) și eliberăm spațiul corespunzător nodului **albastru**, anulând și pointerul care ducea la nodul **albastru** (pointerul **verde**)
- Modificările sunt încercuite în arborele obținut


A B C ... P ... Z [
A B C ... R ... Z [
A B C ... I ... Z [
A B C ... G ... Z [
A B C ... G ... Z [
A B C ... N ... Z [
A ... Z [

Observații

- Problema se prezintă recursiv la nivelul nodului **verde**
- Se anulează unul din pointerii din nodul **verde** (pointerul corespunzător caracterului ‘N’)

Observații

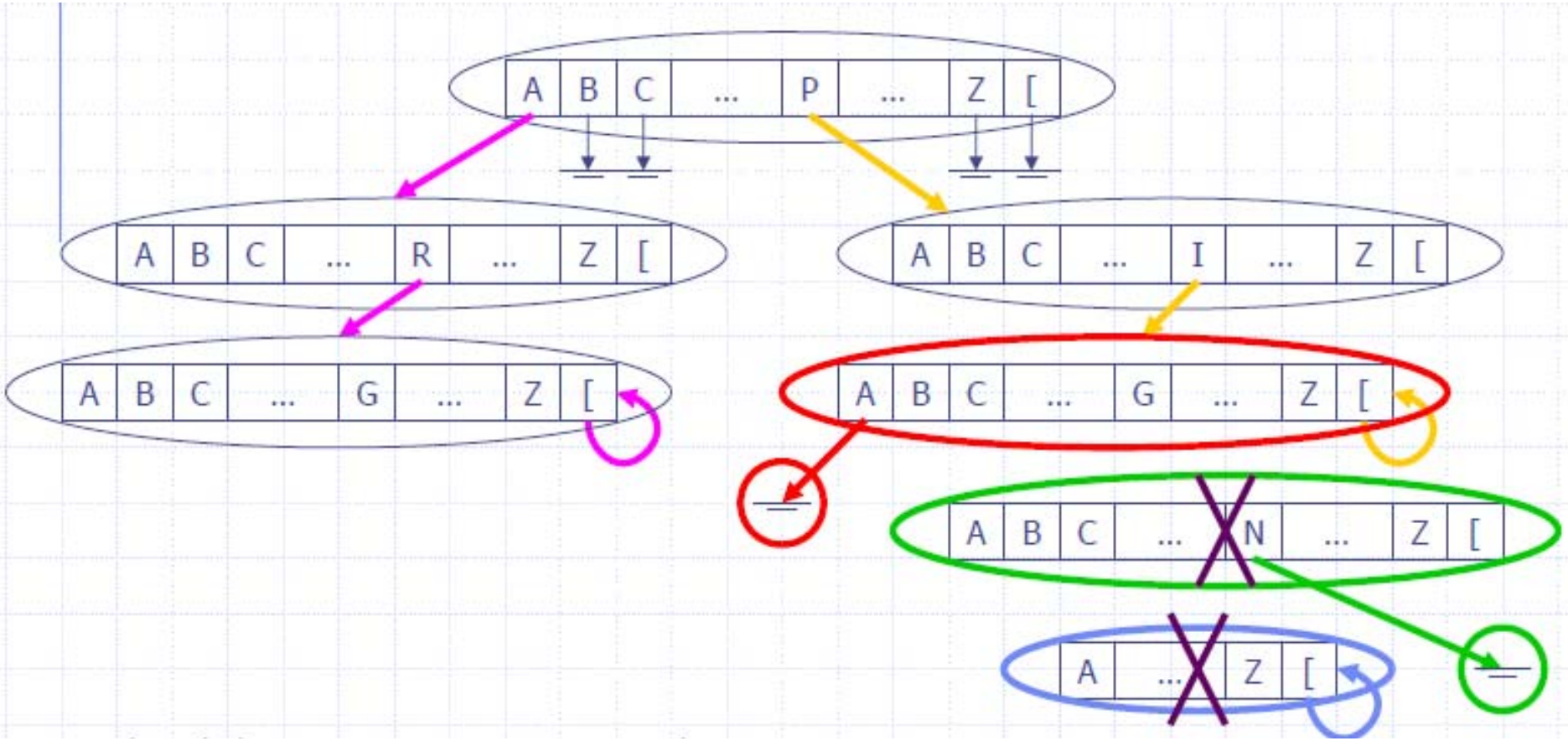
- Deoarece nodul curent (nodul **verde**) nu mai conține pointeri nenuli (toți sunt nuli), înseamnă că nu mai există cuvinte care încep cu prefixul “PIA”, deci nu are rost să menținem nici spațiul ocupat pentru nodul **verde**

Observații

- Acesta va fi eliberat în aceeași manieră și procesul continuă în nodul părinte
- Se va opri abia atunci când, în urma anulării unui pointer din cadrul unui nod, în acel nod mai rămân pointeri nenuli – această situație “salvează” nodul respectiv de la a fi dezalocat

Observații

- Rezultatul ștergerii cuvântului PIAN este:



Concluzii

- Structura de arbore de regăsire este utilizată pentru memorarea mulțimilor de cuvinte
- Într-o mulțime obișnuită de cuvinte inserarea, ștergerea sau căutarea de cuvinte sunt operații cu performanțe proporționale cu numărul total de cuvinte din mulțime (tablou neordonat, listă înlănțuită) sau cel mult cu logaritmul numărului total de cuvinte din mulțime (tablou ordonat, arbore binar de căutare)

Concluzii

- Într-un arbore de regăsire inserarea, ștergerea sau căutarea unui cuvânt sunt operații ale căror performanțe depind **doar** de lungimea cuvântului, adică **nu depind deloc** de numărul de cuvinte care există deja în mulțime
- Dezavantajul arborilor de regăsire este că ocupă un spațiu de memorie destul de mare și în mare parte nefolosit, mulți pointeri fiind nuli

Concluzii

- Dacă setul de cuvinte memorat în cadrul mulțimii are o distribuție omogenă pe lungimea setului de caractere (de exemplu, cuvintele dintr-o limbă formează un set relativ omogen – numărul de cuvinte care încep cu litera ‘A’ este comparabil cu numărul de cuvinte care încep cu litera ‘B’ sau ‘C’, etc.), atunci dezavantajul memoriei se manifestă mai puțin, în sensul că vor exista mult mai puțini pointeri nefolosiți, în raport cu numărul total de pointeri ai structurii