(BP) Balanced Parentheses

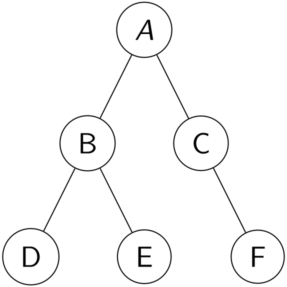
Olga Scheftelowitsch
Marvin Rensing
Frederik Stehli

Pre-order traversal

1. Knoten
2. Linker Teilbaum
3. Rechter Teilbaum

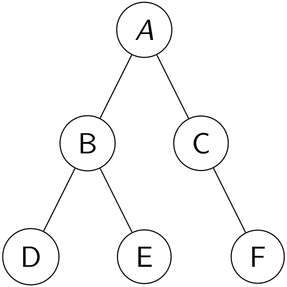
Pre-order traversal

1. Knoten
2. Linker Teilbaum
3. Rechter Teilbaum



Pre-order traversal

1. Knoten
2. Linker Teilbaum
3. Rechter Teilbaum



A B D E C F

Gewinnung von BP Sets

- Durchlaufe den Baum (Pre-order)

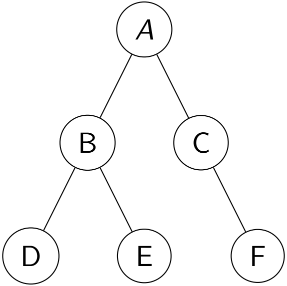
Gewinnung von BP Sets

- Durchlaufe den Baum (Pre-order)
- Neuer Knoten $\rightarrow$ "("

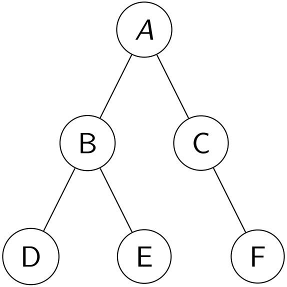
Gewinnung von BP Sets

- Durchlaufe den Baum (Pre-order)
- Neuer Knoten $\rightarrow$ "("
- Knoten abgearbeitet $\rightarrow$ ")"

Gewinnung von BP Sets

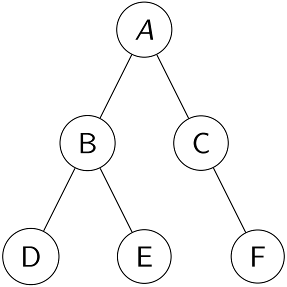


Gewinnung von BP Sets



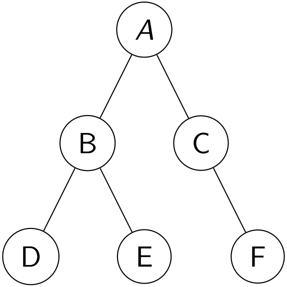
P=(

Gewinnung von BP Sets



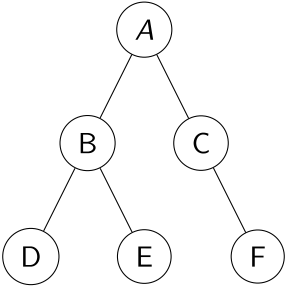
P=((

Gewinnung von BP Sets



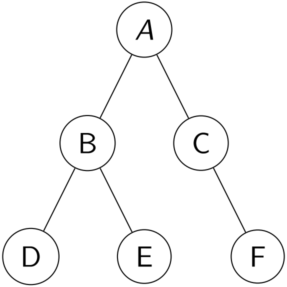
P=(((

Gewinnung von BP Sets



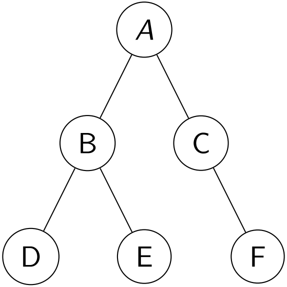
P=((( )

Gewinnung von BP Sets



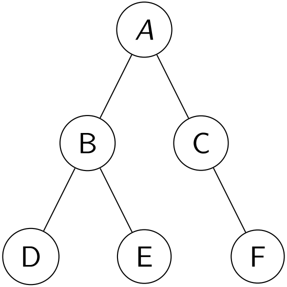
P=((( ) (

Gewinnung von BP Sets



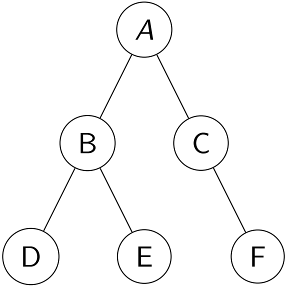
P=((( ) ( )

Gewinnung von BP Sets



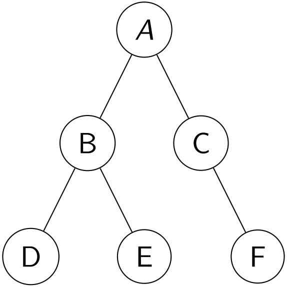
P=((( ) ( ))

Gewinnung von BP Sets



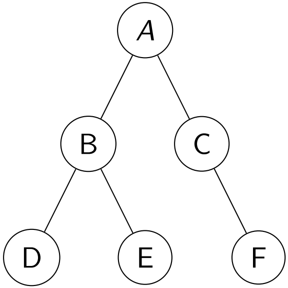
P=((( ) ( )) (

Gewinnung von BP Sets



P=((( ) ( )) (( )))

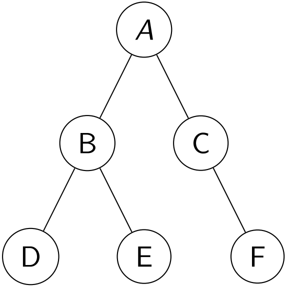
Gewinnung von BP Sets



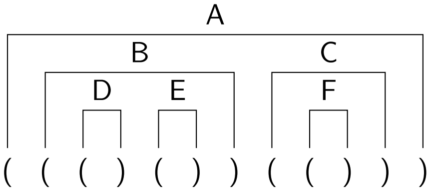
P=((( ) ( )) (( )))

((( ) ( )) (( )))

Gewinnung von BP Sets



P=((( ) ( )) (( )))

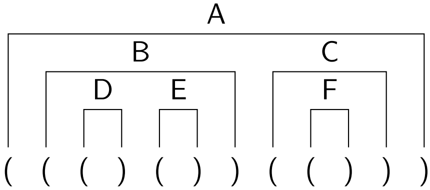


pre_rank und *pre_select*

- $pre\_rank(x) = rank_{(}(x)$
- $pre\_select(p) = select_{(}(p)$

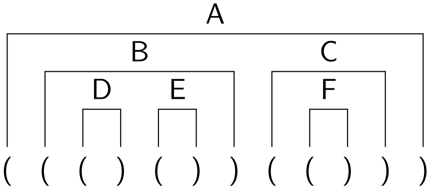
isleaf(x)

- Durchlaufe den Baum (Pre-order)
- Neuer Knoten $\rightarrow$ "("
- Knoten abgearbeitet $\rightarrow$ ")"



isleaf(x)

- Durchlaufe den Baum (Pre-order)
- Neuer Knoten $\rightarrow$ "("
- Knoten abgearbeitet $\rightarrow$ ")"
- $P[x+1] =$")"



findclose(x)

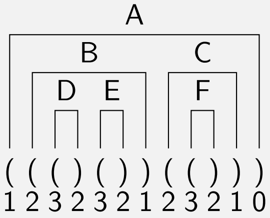
- Index schließende Klammer von Knoten x

findclose(x)

- Index schließende Klammer von Knoten x

excess(x)

- Anzahl nicht geschlossener Klammern bis Position x
- $excess(x) = rank_((x) - rank_)(x)$

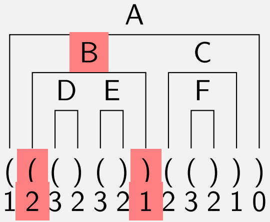


findclose(x)

- Index schließende Klammer von Knoten x

excess(x)

- Anzahl nicht geschlossener Klammern bis Position x
- $excess(x) = rank_{(}(x) - rank_{)}(x)$



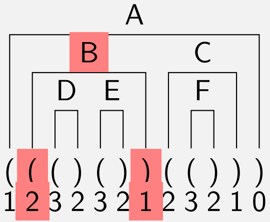
- *excess* schließende Klammer = *excess* öffnende -1

findclose(x)

- Index schließende Klammer von Knoten x

$excess(x)$

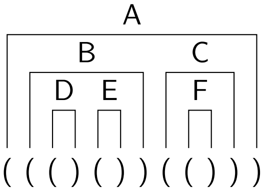
- Anzahl nicht geschlossener Klammern bis Position x
- $excess(x) = rank_{(}(x) - rank_{)}(x)$



- *excess* schließende Klammer $=$ *excess* öffnende -1
- $findclose(x) = min\{y > x \mid excess(y) = excess(x) - 1\} = fwdsearch(x, -1)$

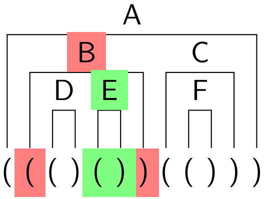
$ancestor(x, y)$

- Ist Knoten x ein Vorfahre von Knoten y.



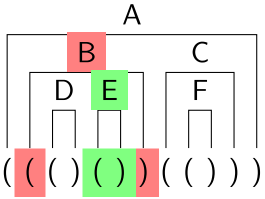
ancestor(x, y)

- Ist Knoten x ein Vorfahre von Knoten y.



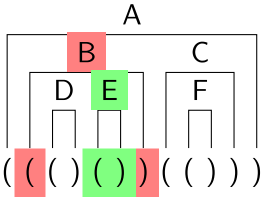
$ancestor(x, y)$

- Ist Knoten x ein Vorfahre von Knoten y.
- Informel: Knoten y von Klammern von x eingeschlossen



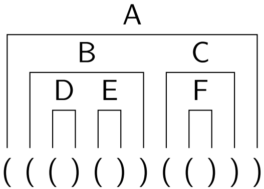
$ancestor(x, y)$

- Ist Knoten x ein Vorfahre von Knoten y.
- Informel: Knoten y von Klammern von x eingeschlossen
- $ancestor(x, y) = x \leq y \leq findclose(x)$



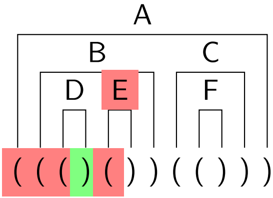
$depth(x)$

- Tiefe des Knoten x



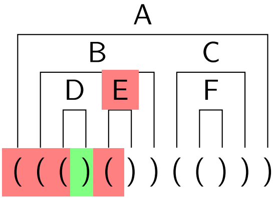
depth(x)

- Tiefe des Knoten x



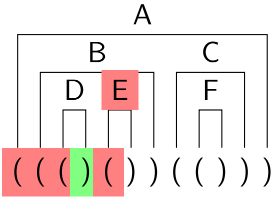
depth(x)

- Tiefe des Knoten x
- Informel: Anzahl nicht geschlossener Klammern links von x



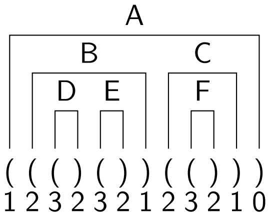
depth(x)

- Tiefe des Knoten x
- Informel: Anzahl nicht geschlossener Klammern links von x
- $depth(x) = rank_{(}(x) - rank_{)}(x) = excess(x)$



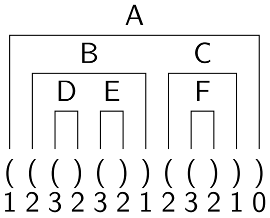
$parent(x)$

- Elternknoten von Knoten x



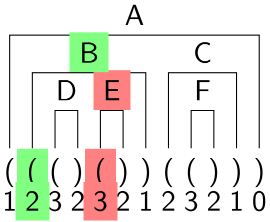
parent(x)

- Elternknoten von Knoten x
- Informel: Naheste öffnende Klammer, die x umschließt



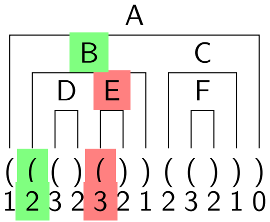
parent(x)

- Elternknoten von Knoten x
- Informel: Naheste öffnende Klammer, die x umschließt



parent(x)

- Elternknoten von Knoten x
- Informel: Naheste öffnende Klammer, die x umschließt
- $\mathit{parent}(x) = \mathit{bwdsearch}(x, -2) + 1 = \mathit{enclose}(x)$



$first\_child(x)$

- Position des ersten Kindes von x

first_child(x)

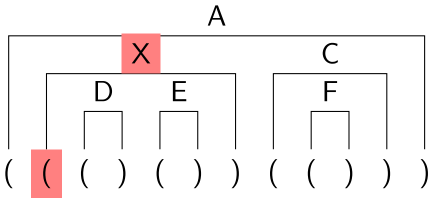
- Position des ersten Kindes von x
- Da das erste Kind des Knotens x beim preorder-Durchlauf sofort danach entdeckt wird, ist seine '(' direkt nach der von Knoten x

$first\_child(x)$

- Position des ersten Kindes von x
- Da das erste Kind des Knotens x beim preorder-Durchlauf sofort danach entdeckt wird, ist seine '(' direkt nach der von Knoten x
- $first\_child(x) = x + 1$

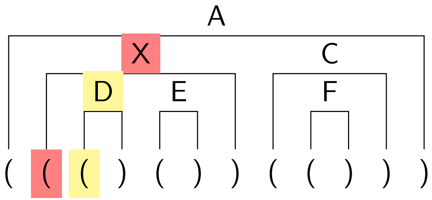
first_child(x)

- Position des ersten Kindes von x
- Da das erste Kind des Knotens x beim preorder-Durchlauf sofort danach entdeckt wird, ist seine '(' direkt nach der von Knoten x
- $first\_child(x) = x + 1$



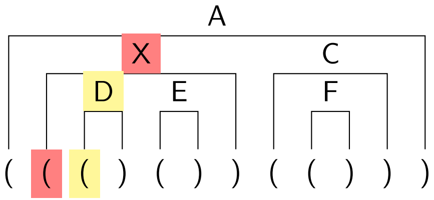
first_child(x)

- Position des ersten Kindes von x
- Da das erste Kind des Knotens x beim preorder-Durchlauf sofort danach entdeckt wird, ist seine '(' direkt nach der von Knoten x
- $first\_child(x) = x + 1$



$first\_child(x)$

- Position des ersten Kindes von x
- Da das erste Kind des Knotens x beim preorder-Durchlauf sofort danach entdeckt wird, ist seine '(' direkt nach der von Knoten x
- $first\_child(x) = x + 1$



- Ausnahme:
  An der stelle $x + 1$ befindet sich ')', den dann hat x keine Kinder und ist ein Blatt

$next\_sibling(x)$

- Position des nächsten Geschwisterknotens von x

$next\_sibling(x)$

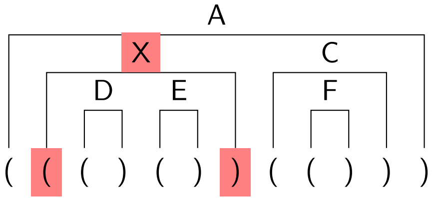
- Position des nächsten Geschwisterknotens von x
- Nachdem der Unterbaum eines Kindknotens abgearbeitet wurde wird seine ')' gesetzt, der nächste Kindknoten besucht und seine '(' gesetzt

$next\_sibling(x)$

- Position des nächsten Geschwisterknotens von x
- Nachdem der Unterbaum eines Kindknotens abgearbeitet wurde wird seine ')' gesetzt, der nächste Kindknoten besucht und seine '(' gesetzt
- Somit gilt: $next\_sibling(x) = findclose(x) + 1$

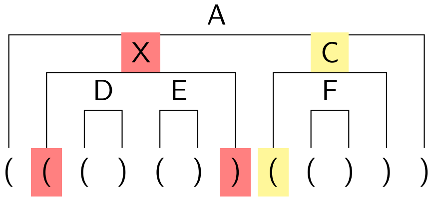
$next\_sibling(x)$

- Position des nächsten Geschwisterknotens von x
- Nachdem der Unterbaum eines Kindknotens abgearbeitet wurde wird seine ')' gesetzt, der nächste Kindknoten besucht und seine '(' gesetzt
- Somit gilt: $next\_sibling(x) = findclose(x) + 1$



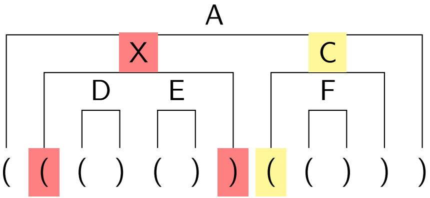
next_sibling(x)

- Position des nächsten Geschwisterknotens von x
- Nachdem der Unterbaum eines Kindknotens abgearbeitet wurde wird seine ')' gesetzt, der nächste Kindknoten besucht und seine '(' gesetzt
- Somit gilt: $next\_sibling(x) = findclose(x) + 1$



next_sibling(x)

- Position des nächsten Geschwisterknotens von x
- Nachdem der Unterbaum eines Kindknotens abgearbeitet wurde wird seine ')' gesetzt, der nächste Kindknoten besucht und seine '(' gesetzt
- Somit gilt: $\mathit{next\_sibling}(x) = \mathit{findclose}(x) + 1$



- Ausnahme:
  An der stelle $\mathit{findclose}(x) + 1$ befindet sich ')', den dann hat x keine weiteren Geschwister und der Unterbaum des Elternknotens ist fertig

$subtree\_size(x)$

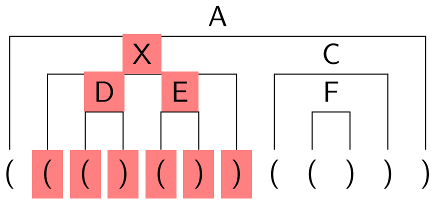
- Anzahl an Knoten im subtree von x

subtree_size(x)

- Anzahl an Knoten im subtree von x
- Alle Knoten des subtrees von x befinden sich in den Klammern von x (einschließlich dieser)

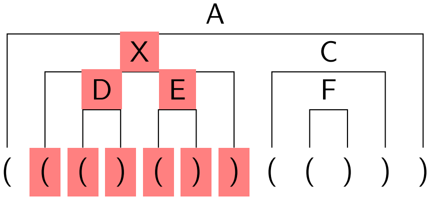
subtree_size(x)

- Anzahl an Knoten im subtree von x
- Alle Knoten des subtrees von x befinden sich in den Klammern von x (einschließlich dieser)



subtree_size(x)

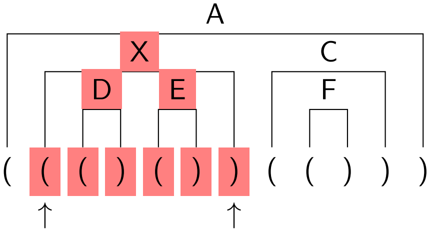
- Anzahl an Knoten im subtree von x
- Alle Knoten des subtrees von x befinden sich in den Klammern von x (einschließlich dieser)



- Alle Knoten haben zwei Klammern

subtree_size(x)

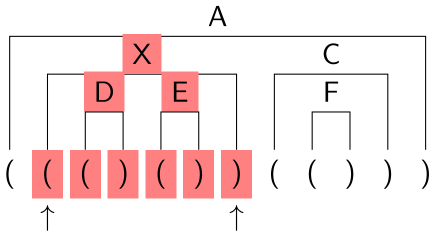
- Anzahl an Knoten im subtree von x
- Alle Knoten des subtrees von x befinden sich in den Klammern von x (einschließlich dieser)



- Alle Knoten haben zwei Klammern

subtree_size(x)

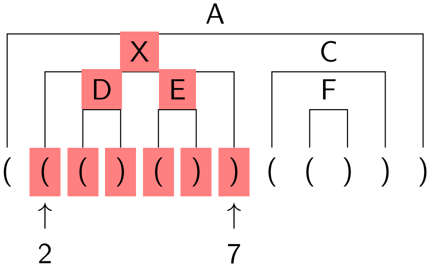
- Anzahl an Knoten im subtree von x
- Alle Knoten des subtrees von x befinden sich in den Klammern von x (einschließlich dieser)



- Alle Knoten haben zwei Klammern
- $\rightarrow$ Die Anzahl der Knoten im Subtree von x ist die Hälfte der Zeichen innerhalb der Klammern von x (einschließlich dieser)

subtree_size(x)

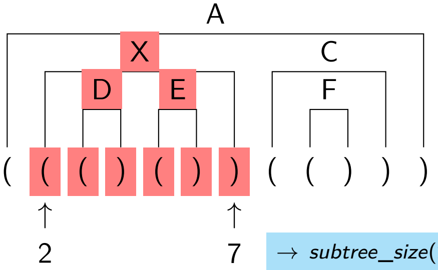
- Anzahl an Knoten im subtree von x
- Alle Knoten des subtrees von x befinden sich in den Klammern von x (einschließlich dieser)



- Alle Knoten haben zwei Klammern
- $\rightarrow$ Die Anzahl der Knoten im Subtree von x ist die Hälfte der Zeichen innerhalb der Klammern von x (einschließlich dieser)

subtree_size(x)

- Anzahl an Knoten im subtree von x
- Alle Knoten des subtrees von x befinden sich in den Klammern von x (einschließlich dieser)

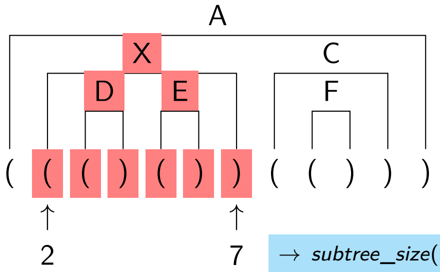


$\rightarrow$ *subtree_size*$(2) = 3 = 6/2 = (7 - 2 + 1)/2$

- Alle Knoten haben zwei Klammern
- $\rightarrow$ Die Anzahl der Knoten im Subtree von x ist die Hälfte der Zeichen innerhalb der Klammern von x (einschließlich dieser)

subtree_size(x)

- Anzahl an Knoten im subtree von x
- Alle Knoten des subtrees von x befinden sich in den Klammern von x (einschließlich dieser)



$\rightarrow$ $\mathit{subtree\_size}(2) = 3 = 6/2 = (7 - 2 + 1)/2$

- Alle Knoten haben zwei Klammern
- $\rightarrow$ Die Anzahl der Knoten im Subtree von x ist die Hälfte der Zeichen innerhalb der Klammern von x (einschließlich dieser)
- $\rightarrow$ $\mathit{subtree\_size}(x) = (\mathit{findclose}(x) - x + 1)/2$