

Optimale Bearbeitung von XML-Anfragen

Volker Grabsch Christine Janischek

4. Februar 2008

Allgemeines

- veröffentlicht unter
<http://www.prof.v.de/uni/>

- lizenziert unter



Creative Commons BY-SA 3.0

Quelle

Dieser Vortrag basiert auf dem Paper

Holistic Twig Joins: Optimal XML Pattern Matching

von Nicolas Bruno, Nick Koudas und Divesh Srivastava.

Übersicht

- 1 Einleitung
- 2 XML-Darstellung
- 3 PathStack
- 4 Algorithmen

Übersicht

- 1 **Einleitung**
- 2 XML-Darstellung
- 3 PathStack
- 4 Algorithmen

Einleitung

- Ziel: Bessere Algorithmen zur XML-Anfragenabarbeitung
- XPath-Anfragen auf XML-Datenbank
- Problem: unverhältnismässig viele Teilergebnisse bei stark einschränkenden Anfragen

Übersicht

- 1 Einleitung
- 2 XML-Darstellung**
- 3 PathStack
- 4 Algorithmen

XML-Darstellung

- bisher
 - ParentID
 - Dewey-Pos
 - OrdPath
 - Preorder : Postorder
 - usw.
- jetzt
 - (DocID, LeftPos : RightPos, LevelNum)
 - sehr ähnlich zu Preorder : Postorder

XML-Darstellung

- bisher
 - ParentID
 - Dewey-Pos
 - OrdPath
 - Preorder : Postorder
 - usw.
- jetzt
 - (DocID, LeftPos : RightPos, LevelNum)
 - sehr ähnlich zu Preorder : Postorder

LeftPos : RightPos

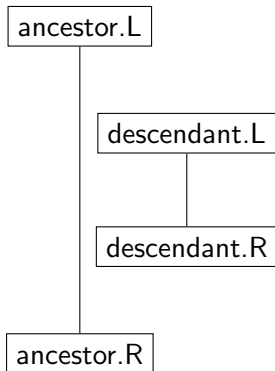
```

1 <report>
2   <title>
3     Der moderne Dieb
4   </title>
5   <section>
6     <title>
7       Unterwegs
8     </title>
9     <content>
10      <text>
11        Gerade zu Weihnachten ...
12      </text>
13      <section>
14        ...
22      </section>
23      <section>
24        ...
32      </section>
33    </content>
34  </section>
35 </report>

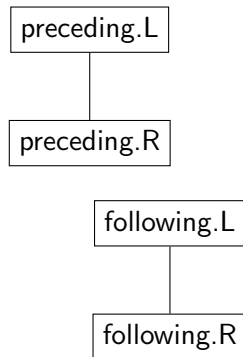
```

LeftPos : RightPos	
1:35	<report>
2:4	<title>
3:3	Der moderne ...
5:34	<section>
6:8	<title>
7:7	Unterwegs
9:33	<content>
10:12	<text>
11:11	Gerade zu W...
13:22	<section>
...	
23:32	<section>
...	

Achsen mit (LeftPos : RightPos)



ancestor.L < descendant.L
ancestor.R > descendant.R



preceding.R < following.L

Übersicht

- 1 Einleitung
- 2 XML-Darstellung
- 3 PathStack**
- 4 Algorithmen

Grundlagen

- Anfrage-Baum (XPath-Achsen)
- Filter auf Anfrage-Knoten (Tagname, XPath-Prädikate)
- Eingabe: Kandidaten für Anfrage-Knoten (in preorder)

```
//section//title[text()!='']
```



Grundlagen

- Anfrage-Baum (XPath-Achsen)
- Filter auf Anfrage-Knoten (Tagname, XPath-Prädikate)
- Eingabe: Kandidaten für Anfrage-Knoten (in preorder)

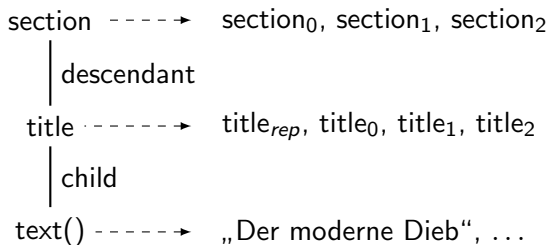
```
//section//title[text()!='']
```



Grundlagen

- Anfrage-Baum (XPath-Achsen)
- Filter auf Anfrage-Knoten (Tagname, XPath-Prädikate)
- Eingabe: Kandidaten für Anfrage-Knoten (in preorder)

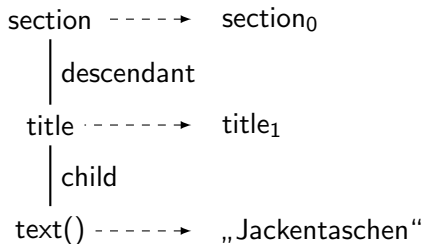
```
//section//title[text()!='']
```



Grundlagen

- Anfrage-Baum (XPath-Achsen)
- Filter auf Anfrage-Knoten (Tagname, XPath-Prädikate)
- Eingabe: Kandidaten für Anfrage-Knoten (in preorder)
- Ausgabe: Zuordnungen Dokument-Baum → Anfrage-Baum

```
//section//title[text()!='']
```



root-to-leaf \leftrightarrow leaf-to-root

```
//section//title[text()!='']
```

section	title	text()	
5:34	6:8	7:7	/r./section ₀ /title ₀ /„Unterwegs“
5:34	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
5:34	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“
13:22	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
23:32	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“

section	title	text()	
5:34	6:8	7:7	/r./section ₀ /title ₀ /„Unterwegs“
5:34	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
13:22	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
5:34	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“
23:32	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“

root-to-leaf \leftrightarrow leaf-to-root

```
//section//title[text()=' ']
```

section	title	text()	
5:34	6:8	7:7	/r./section ₀ /title ₀ /„Unterwegs“
5:34	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
5:34	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“
13:22	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
23:32	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“

section	title	text()	
5:34	6:8	7:7	/r./section ₀ /title ₀ /„Unterwegs“
5:34	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
13:22	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
5:34	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“
23:32	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“

Erkenntnis

- Reihenfolge leaf-to-root effizienter erzeugbar
- pro Dokumenten-Blatt
 - alle möglichen Matches seines Pfads durchlaufen
 - kompakte Kodierung durch Stacks

Erkenntnis

- Reihenfolge leaf-to-root effizienter erzeugbar
- pro Dokumenten-Blatt
 - alle möglichen Matches seines Pfads durchlaufen
 - kompakte Kodierung durch Stacks

Primitive Stacks

```
//section//title[text()!='']
```

section	title	text()	
5:34	6:8	7:7	/r./section ₀ /title ₀ /„Unterwegs“
5:34	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
13:22	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
5:34	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“
23:32	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“

	Stack	Stack	Stack
section	section ₀	section ₁ , section ₀	section ₂ , section ₀
title	title ₀	title ₁	title ₂
text()	Unterwegs	Jackentaschen	Handtaschen

Primitive Stacks

```
//section//title[text()!='']
```

section	title	text()	
5:34	6:8	7:7	/r./section ₀ /title ₀ /„Unterwegs“
5:34	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
13:22	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
5:34	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“
23:32	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“

	Stack	Stack	Stack
section	section ₀	section ₁ , section ₀	section ₂ , section ₀
title	title ₀	title ₁	title ₂
text()	Unterwegs	Jackentaschen	Handtaschen

Primitive Stacks

```
//section//title[text()!='']
```

section	title	text()	
5:34	6:8	7:7	/r./section ₀ /title ₀ /„Unterwegs“
5:34	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
13:22	14:16	15:15	/r./section ₀ /c./section ₁ /title ₁ /„Jackentaschen“
5:34	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“
23:32	24:26	25:25	/r./section ₀ /c./section ₂ /title ₂ /„Handtaschen“

	Stack	Stack	Stack
section	section ₀	section ₁ , section ₀	section ₂ , section ₀
title	title ₀	title ₁	title ₂
text()	Unterwegs	Jackentaschen	Handtaschen

Primitive Stacks - Problemfall

```
//section//content//text [text()='Jacken werden...']
```

section	content	
5:34	9:33	/r./section ₀ /content ₀ /section ₁ /content ₁ /text ₁
13:22	9:33	/r./section ₀ /content ₀ /section ₁ /content ₁ /text ₁
5:34	17:21	/r./section ₀ /content ₀ /section ₁ /content ₁ /text ₁
13:22	17:21	/r./section ₀ /content ₀ /section ₁ /content ₁ /text ₁

?!

	Stack
section	section ₁ , section ₀
content	content ₁ , content ₀
text	text ₁

Erkenntnis

- Stacks bisher zu primitiv
- Lösungen:
 - prüfe vor Ausgabe auf ancestor-descendant (ineffizient)
 - Hierarchie-Info im Stack notieren

Erkenntnis

- Stacks bisher zu primitiv
- Lösungen:
 - prüfe vor Ausgabe auf ancestor-descendant (ineffizient)
 - Hierarchie-Info im Stack notieren

Erkenntnis

- Stacks bisher zu primitiv
- Lösungen:
 - prüfe vor Ausgabe auf ancestor-descendant (ineffizient)
 - Hierarchie-Info im Stack notieren

Erkenntnis

- Stacks bisher zu primitiv
- Lösungen:
 - prüfe vor Ausgabe auf ancestor-descendant (ineffizient)
 - Hierarchie-Info im Stack notieren

Stacks mit Hierarchie-Info

```
//section//content//text [text()='Jacken werden...']
```

section	content	
5:34	9:33	/r./section ₀ /content ₀ /section ₁ /content ₁ /text ₁
5:34	17:21	/r./section ₀ /content ₀ /section ₁ /content ₁ /text ₁
13:22	17:21	/r./section ₀ /content ₀ /section ₁ /content ₁ /text ₁

	Stack
section	section ₁ , section ₀
content	content ₁ (section ₁), content ₀ (section ₀)
text	text ₁ (content ₁)

PathStack-Algorithmus

- q_{min} := Anfrage-Knoten mit minimalem Dok-Knoten d_{min}
 - gibt es mehrere q_{min} , wähle kleinsten
- säubere alle Stacks
 - entferne alle precedings (d.h. nicht-ancestors) von d_{min}
- $d_{min} \rightarrow q_{min}$ -Stack
- nächster Dok-Knoten für q_{min}
- Wenn q_{min} ein Blatt:
 - Ausgabe der in den Stacks kodierte Lösungen
- wiederhole von vorn

PathStack-Algorithmus

- q_{min} := Anfrage-Knoten mit minimalem Dok-Knoten d_{min}
 - gibt es mehrere q_{min} , wähle kleinsten
- säubere alle Stacks
 - entferne alle `precedings` (d.h. nicht-ancestors) von d_{min}
- $d_{min} \rightarrow q_{min}$ -Stack
- nächster Dok-Knoten für q_{min}
- Wenn q_{min} ein Blatt:
 - Ausgabe der in den Stacks kodierten Lösungen
- wiederhole von vorn

PathStack-Algorithmus

- q_{min} := Anfrage-Knoten mit minimalem Dok-Knoten d_{min}
 - gibt es mehrere q_{min} , wähle kleinsten
- säubere alle Stacks
 - entferne alle precedings (d.h. nicht-ancestors) von d_{min}
- $d_{min} \rightarrow q_{min}$ -Stack
- nächster Dok-Knoten für q_{min}
- Wenn q_{min} ein Blatt:
 - Ausgabe der in den Stacks kodierte Lösungen
- wiederhole von vorn

PathStack-Algorithmus

- q_{min} := Anfrage-Knoten mit minimalem Dok-Knoten d_{min}
 - gibt es mehrere q_{min} , wähle kleinsten
- säubere alle Stacks
 - entferne alle precedings (d.h. nicht-ancestors) von d_{min}
- $d_{min} \rightarrow q_{min}$ -Stack
- nächster Dok-Knoten für q_{min}
- Wenn q_{min} ein Blatt:
 - Ausgabe der in den Stacks kodierte Lösungen
- wiederhole von vorn

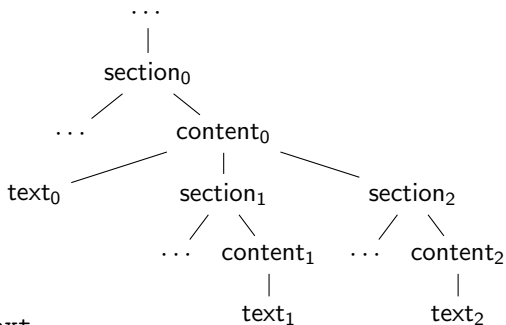
PathStack-Algorithmus

- q_{min} := Anfrage-Knoten mit minimalem Dok-Knoten d_{min}
 - gibt es mehrere q_{min} , wähle kleinsten
- säubere alle Stacks
 - entferne alle precedings (d.h. nicht-ancestors) von d_{min}
- $d_{min} \rightarrow q_{min}$ -Stack
- nächster Dok-Knoten für q_{min}
- Wenn q_{min} ein Blatt:
 - Ausgabe der in den Stacks kodierte Lösungen
- wiederhole von vorn

PathStack-Algorithmus

- q_{min} := Anfrage-Knoten mit minimalem Dok-Knoten d_{min}
 - gibt es mehrere q_{min} , wähle kleinsten
- säubere alle Stacks
 - entferne alle precedings (d.h. nicht-ancestors) von d_{min}
- $d_{min} \rightarrow q_{min}$ -Stack
- nächster Dok-Knoten für q_{min}
- Wenn q_{min} ein Blatt:
 - Ausgabe der in den Stacks kodierten Lösungen
- wiederhole von vorn

Vorführung: PathStack

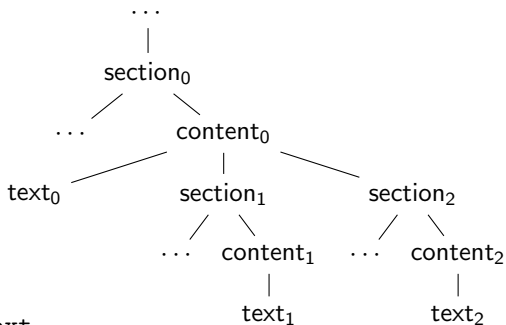


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section	section ₀ , ...	
	content	content ₀ , ...	
	text	text ₀ , ...	

Ausgabe:

Vorführung: PathStack

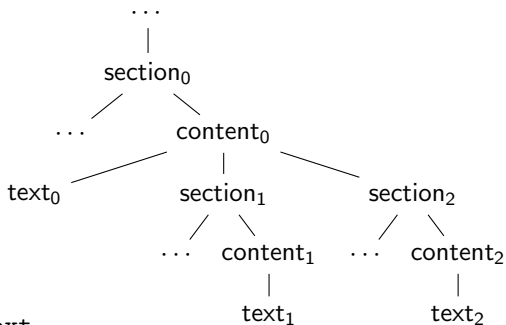


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
→	section	section ₀ , ...	
	content	content ₀ , ...	
	text	text ₀ , ...	

Ausgabe:

Vorführung: PathStack

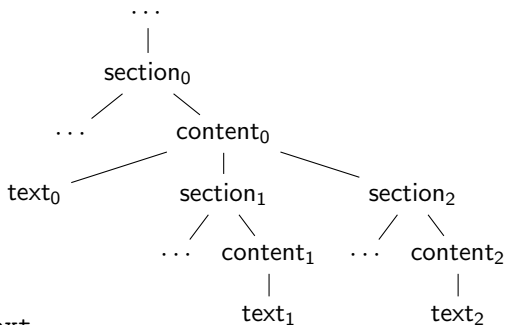


```
//section//content//text
```

q_{min}	q	Dok-Knoten	Stack
→	section	section ₁ , ...	section ₀
	content	content ₀ , ...	
	text	text ₀ , ...	

Ausgabe:

Vorführung: PathStack

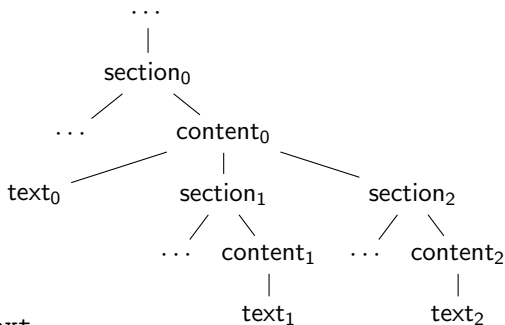


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section	section ₁ , ...	section ₀
→	content	content ₀ , ...	
	text	text ₀ , ...	

Ausgabe:

Vorführung: PathStack

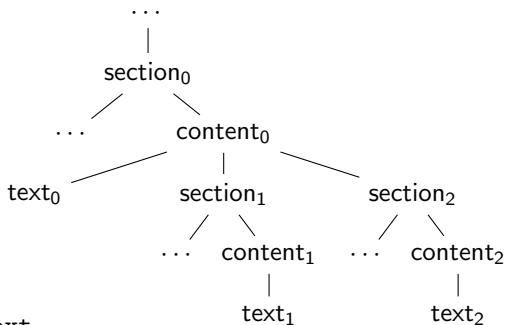


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section	section ₁ , ...	section ₀
→	content	content ₁ , ...	content ₀ (section ₀)
	text	text ₀ , ...	

Ausgabe:

Vorführung: PathStack

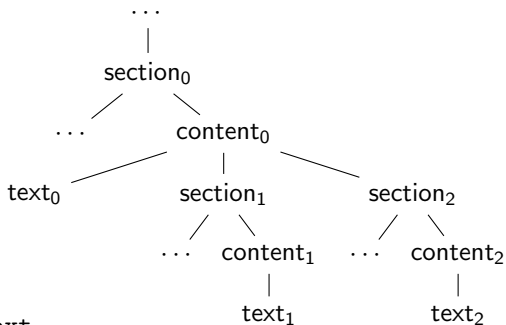


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section	section ₁ , ...	section ₀
	content	content ₁ , ...	content ₀ (section ₀)
→	text	text ₀ , ...	

Ausgabe:

Vorführung: PathStack

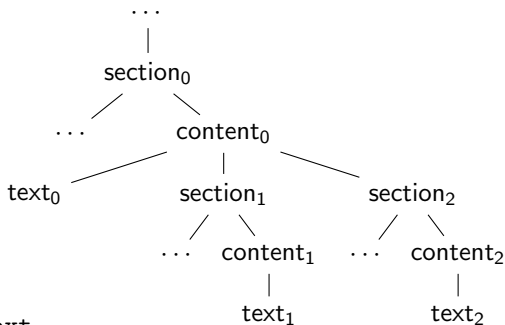


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section	section ₁ , ...	section ₀
	content	content ₁ , ...	content ₀ (section ₀)
→	text	text ₁ , ...	text ₀ (content ₀)

Ausgabe:

Vorführung: PathStack

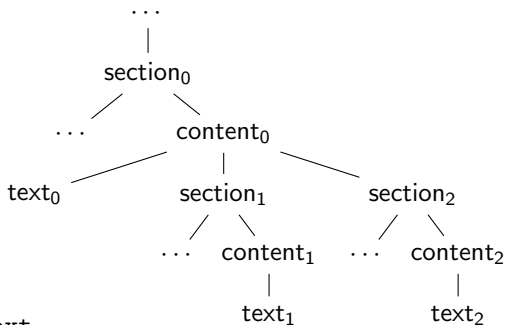


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section	section ₁ , ...	section ₀
	content	content ₁ , ...	content ₀ (section ₀)
→	text	text ₁ , ...	text ₀ (content ₀)

Ausgabe: `/report/section0/content0/text0`

Vorführung: PathStack

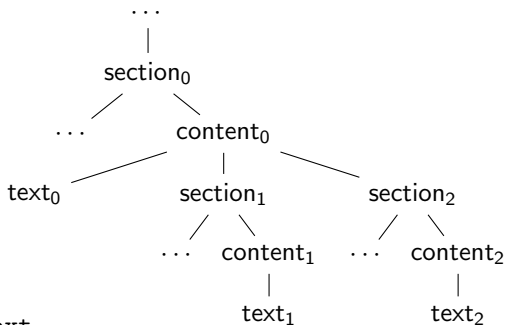


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
→	section	section ₁ , ...	section ₀
	content	content ₁ , ...	content ₀ (section ₀)
	text	text ₁ , ...	text ₀ (content ₀)

Ausgabe:

Vorführung: PathStack

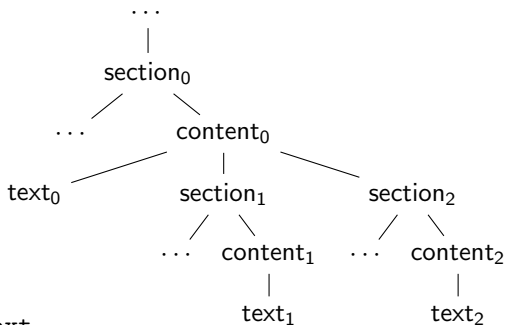


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
→	section	section ₁ , ...	section ₀
	content	content ₁ , ...	content ₀ (section ₀)
	text	text ₁ , ...	

Ausgabe:

Vorführung: PathStack

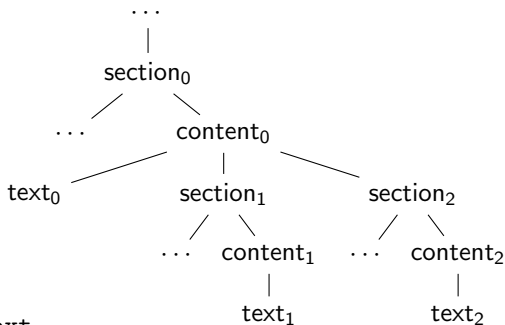


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
\rightarrow	section	section ₂	section ₁ , section ₀
	content	content ₁ , ...	content ₀ (section ₀)
	text	text ₁ , ...	

Ausgabe:

Vorführung: PathStack

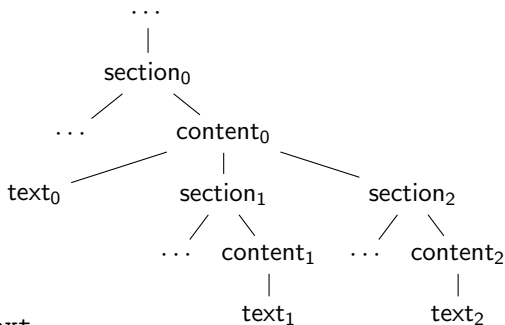


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section	section ₂	section ₁ , section ₀
→	content	content ₁ , ...	content ₀ (section ₀)
	text	text ₁ , ...	

Ausgabe:

Vorführung: PathStack

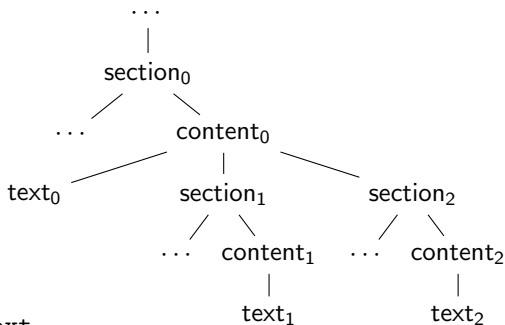


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section	section ₂	section ₁ , section ₀
→	content	content ₂	content ₁ (section ₁), content ₀ (section ₀)
	text	text ₁ , ...	

Ausgabe:

Vorführung: PathStack

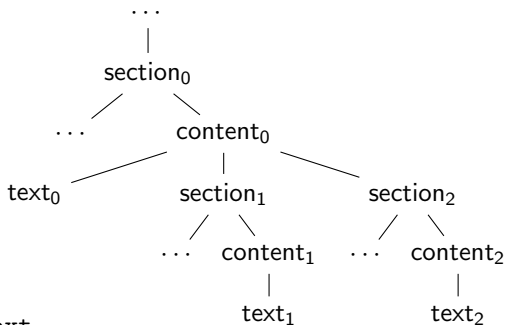


//section//content//text

q_{min}	q	Dok-Knoten	Stack
	section	section ₂	section ₁ , section ₀
	content	content ₂	content ₁ (section ₁), content ₀ (section ₀)
→	text	text ₂	text ₁ (content ₁)

Ausgabe:

Vorführung: PathStack

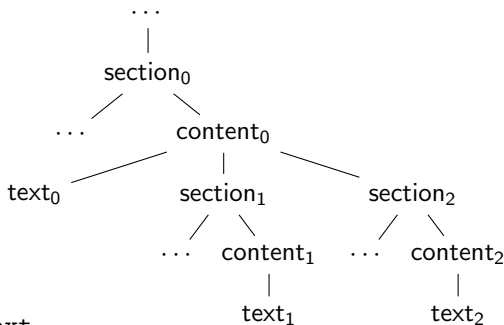


//section//content//text

q_{min}	q	Dok-Knoten	Stack
	section	section ₂	section ₁ , section ₀
	content	content ₂	content ₁ (section ₁), content ₀ (section ₀)
→	text	text ₂	text ₁ (content ₁)

Ausgabe: /report/section₀/content₀/section₁/content₁/t1

Vorführung: PathStack

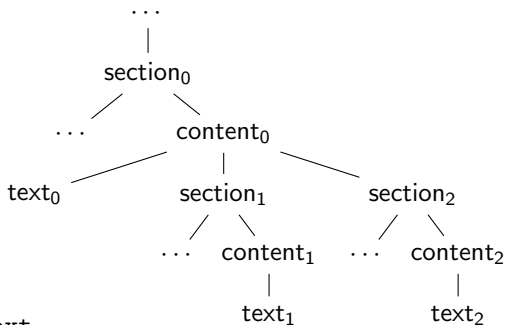


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section	section ₂	section ₁ , section ₀
	content	content ₂	content ₁ (section ₁), content ₀ (section ₀)
→	text	text ₂	text ₁ (content ₁)

Ausgabe: `/report/section0/content0/section1/content1/t1`

Vorführung: PathStack

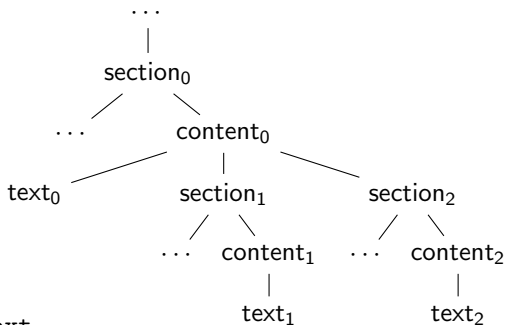


//section//content//text

q_{min}	q	Dok-Knoten	Stack
	section	section ₂	section ₁ , section ₀
	content	content ₂	content ₁ (section ₁), content ₀ (section ₀)
→	text	text ₂	text ₁ (content ₁)

Ausgabe: /report/section₀/content₀/section₁/content₁/t1

Vorführung: PathStack

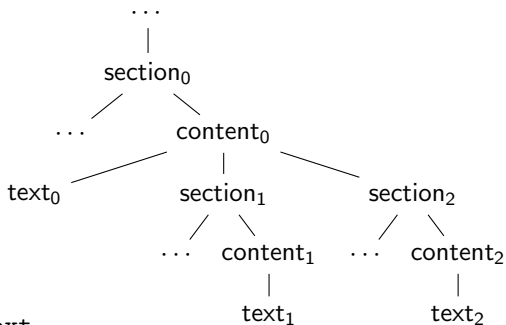


//section//content//text

q_{min}	q	Dok-Knoten	Stack
→	section	section ₂	section ₁ , section ₀
	content	content ₂	content ₁ (section ₁), content ₀ (section ₀)
	text	text ₂	text ₁ (content ₁)

Ausgabe:

Vorführung: PathStack

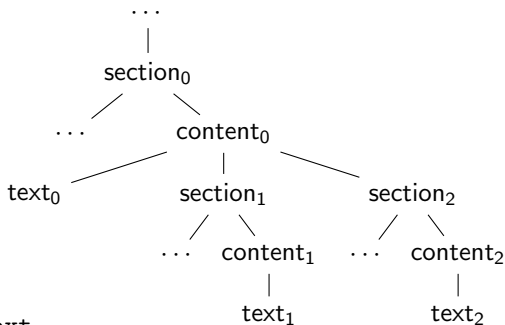


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
→	section	section ₂	section ₀
	content	content ₂	content ₀ (section ₀)
	text	text ₂	

Ausgabe:

Vorführung: PathStack

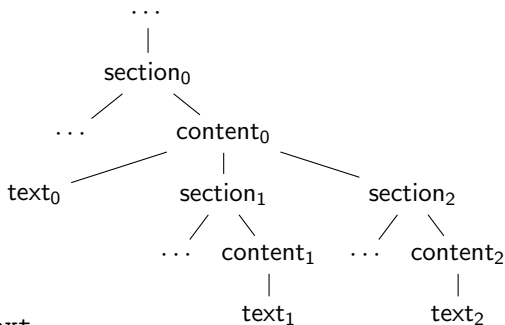


//section//content//text

q_{min}	q	Dok-Knoten	Stack
→	section		section ₂ , section ₀
	content	content ₂	content ₀ (section ₀)
	text	text ₂	

Ausgabe:

Vorführung: PathStack

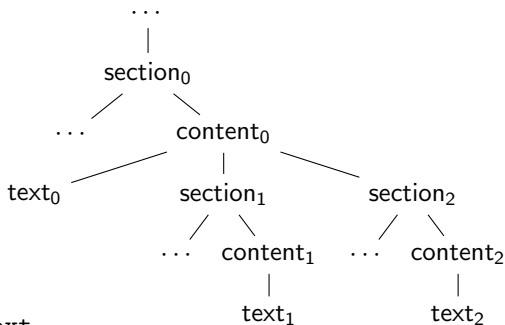


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section		section ₂ , section ₀
→	content	content ₂	content ₀ (section ₀)
	text	text ₂	

Ausgabe:

Vorführung: PathStack

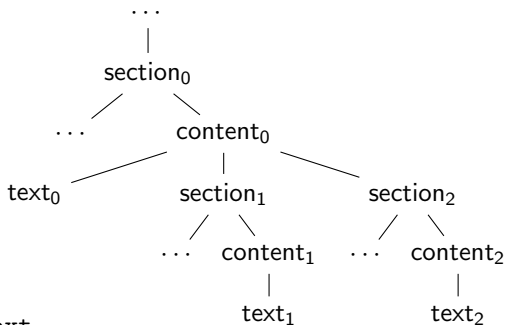


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section		section ₂ , section ₀
→	content		content ₂ (section ₂), content ₀ (section ₀)
	text	text ₂	

Ausgabe:

Vorführung: PathStack

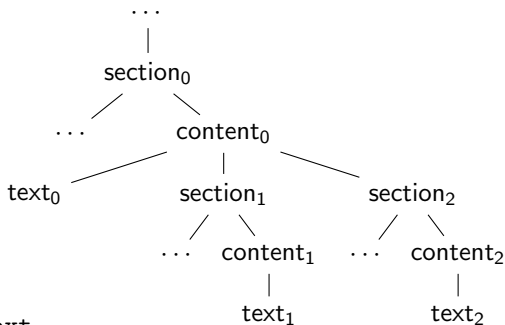


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section		section ₂ , section ₀
	content		content ₂ (section ₂), content ₀ (section ₀)
→	text	text ₂	

Ausgabe:

Vorführung: PathStack

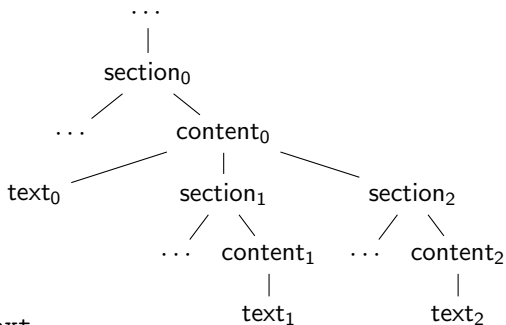


//section//content//text

q_{min}	q	Dok-Knoten	Stack
	section		section ₂ , section ₀
	content		content ₂ (section ₂), content ₀ (section ₀)
→	text		text ₂ (content ₂)

Ausgabe:

Vorführung: PathStack

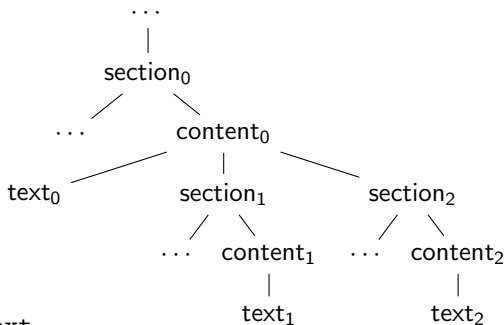


//section//content//text

q_{min}	q	Dok-Knoten	Stack
	section		section ₂ , section ₀
	content		content ₂ (section ₂), content ₀ (section ₀)
→	text		text ₂ (content ₂)

Ausgabe: /report/section₀/content₀/section₂/content₂/t2

Vorführung: PathStack

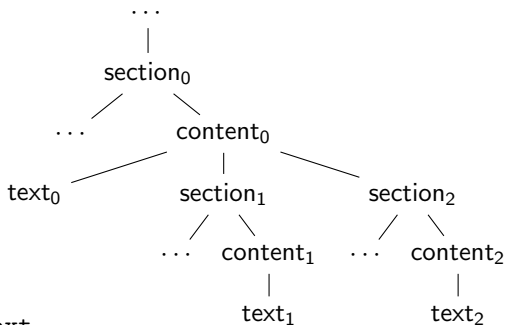


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section		section ₂ , section ₀
	content		content ₂ (section ₂), content ₀ (section ₀)
→	text		text ₂ (content ₂)

Ausgabe: `/report/section0/content0/section2/content2/t2`

Vorführung: PathStack

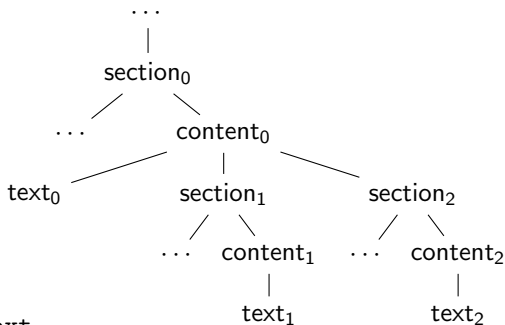


`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
	section		section ₂ , section ₀
	content		content ₂ (section ₂), content ₀ (section ₀)
→	text		text ₂ (content ₂)

Ausgabe: `/report/section0/content0/section2/content2/t2`

Vorführung: PathStack



`//section//content//text`

q_{min}	q	Dok-Knoten	Stack
		section	section ₂ , section ₀
		content	content ₂ (section ₂), content ₀ (section ₀)
→		text	text ₂ (content ₂)

Ausgabe:

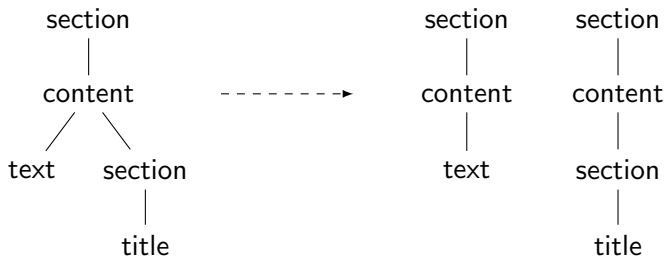
Nach PathStack – Kritik am Paper

- PathStack arbeitet nur auf Zweigen des Anfragebaums ?
- Wende PathStack auf Zweige an, dann Merge der Ausgaben
- vorher: leaf-to-root → root-to-leaf ?
 - Sortierung: ineffizient
 - verzögerte, gepufferte Ausgabe



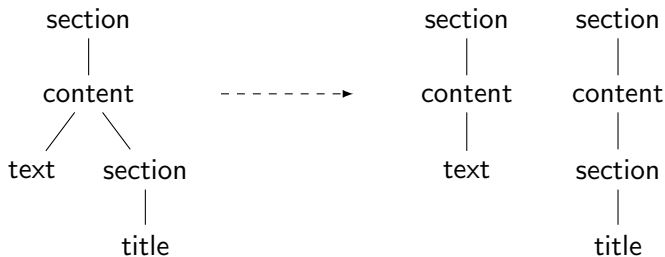
Nach PathStack – Kritik am Paper

- PathStack arbeitet nur auf Zweigen des Anfragebaums ?
- Wende PathStack auf Zweige an, dann Merge der Ausgaben
- vorher: leaf-to-root \rightarrow root-to-leaf ?
 - Sortierung: ineffizient
 - verzögerte, gepufferte Ausgabe



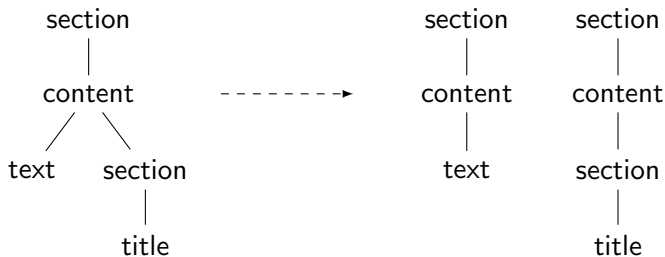
Nach PathStack – Kritik am Paper

- PathStack arbeitet nur auf Zweigen des Anfragebaums ?
- Wende PathStack auf Zweige an, dann Merge der Ausgaben
- vorher: leaf-to-root \rightarrow root-to-leaf ?
 - Sortierung: ineffizient
 - verzögerte, gepufferte Ausgabe



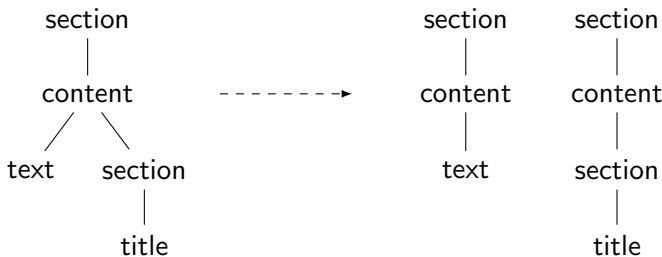
Nach PathStack – Kritik am Paper

- PathStack arbeitet nur auf Zweigen des Anfragebaums ?
- Wende PathStack auf Zweige an, dann Merge der Ausgaben
- vorher: leaf-to-root \rightarrow root-to-leaf ?
 - Sortierung: ineffizient
 - verzögerte, gepufferte Ausgabe



Nach PathStack – Kritik am Paper

- PathStack arbeitet nur auf Zweigen des Anfragebaums ?
- Wende PathStack auf Zweige an, dann Merge der Ausgaben
- vorher: leaf-to-root \rightarrow root-to-leaf ?
 - Sortierung: ineffizient
 - verzögerte, gepufferte Ausgabe



Übersicht

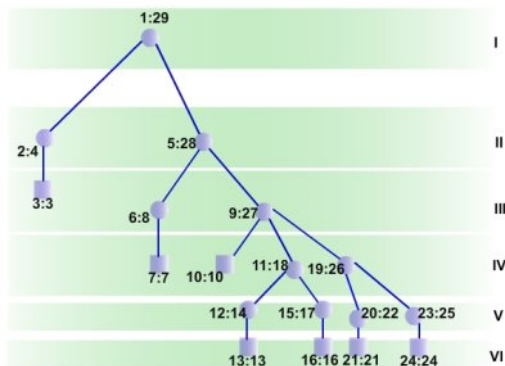
- 1 Einleitung
- 2 XML-Darstellung
- 3 PathStack
- 4 Algorithmen**

Algorithmen

- Multi-Predicate Merge Join (MPMGJN)
 - arbeitet Ancestor/Descendant-Achsen effizient ab
- PathMPMJ
 - Optimierter MPMGJN
 - arbeitet aufeinanderfolgende A/D-Achsen besser ab
- PathStack
 - benutzt zusätzliche Datenstruktur (PathStack)
 - arbeitet aufeinanderfolgende A/D-Achsen optimal ab
- TwigStack
 - basiert auf PathStack
 - arbeitet auch nebeneinander liegende A/D-Achsen optimal ab
- TwigStackXB
 - basiert auf TwigStack
 - sublinear bezüglich der Eingangsdaten

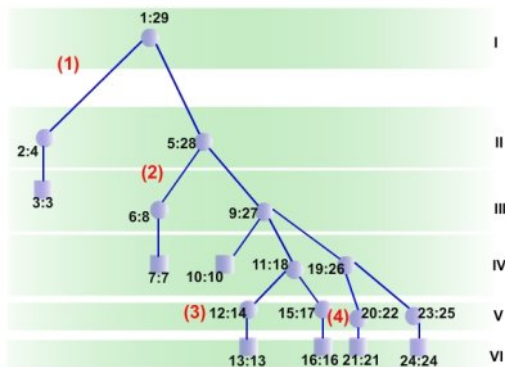
Beispiel(Multi-Predicate Merge Join – MPMGJN/1)

`report[title='unterwegs']/section[title='Jacken%']`



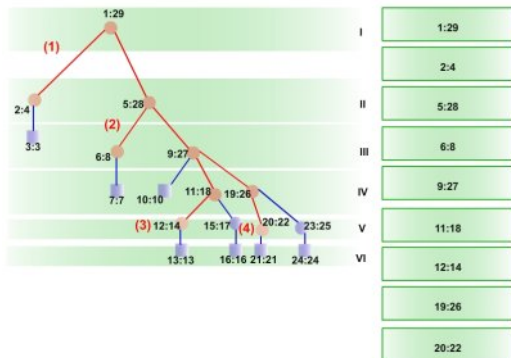
Beispiel(Multi-Predicate Merge Join – MPMGJN/2)

report[title='unterwegs']/section[title='Jacken%']



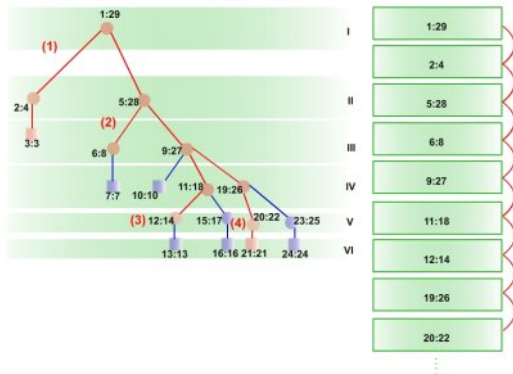
Beispiel(Multi-Predicate Merge Join – MPMGJN/3)

`report[title='unterwegs']/section[title='Jacken%']`



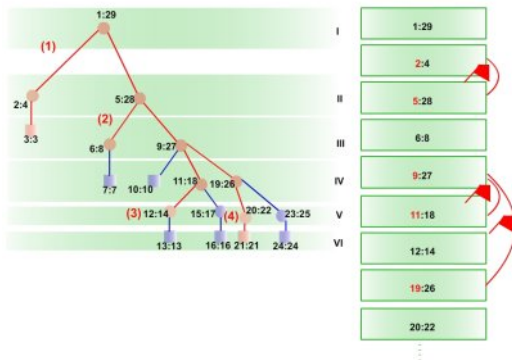
Beispiel(Multi-Predicate Merge Join – MPMGJN/4)

report[title='unterwegs']/section[title='Jacken%']



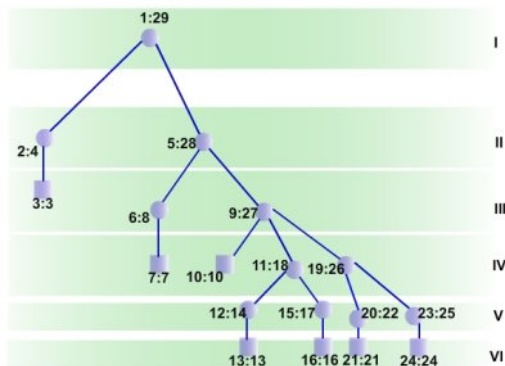
Beispiel(PathMPMJ)

`report[title='unterwegs']/section[title='Jacken%']`



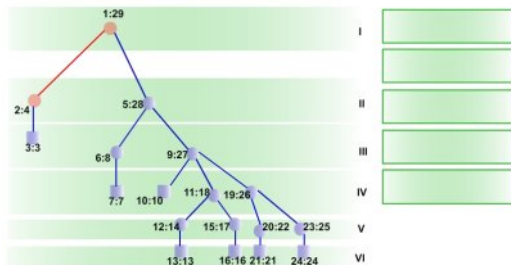
Beispiel(PathStack/1)

`report[title='unterwegs']/section[title='Jacken%']`



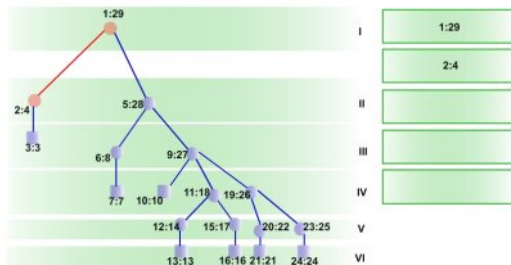
Beispiel(PathStack/2)

`report[title='unterwegs']/section[title='Jacken%']`



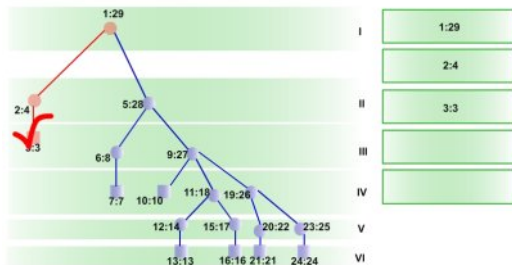
Beispiel(PathStack/3)

`report[title='unterwegs']/section[title='Jacken%']`



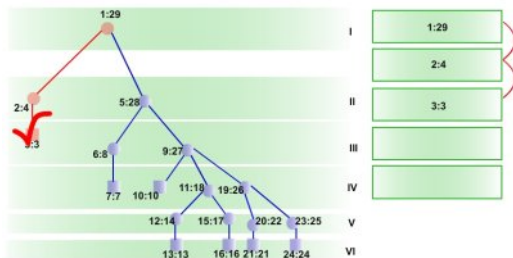
Beispiel(PathStack/4)

`report[title='unterwegs']/section[title='Jacken%']`



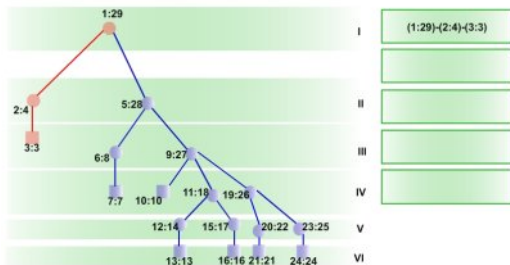
Beispiel(PathStack/5)

`report[title='unterwegs']/section[title='Jacken%']`



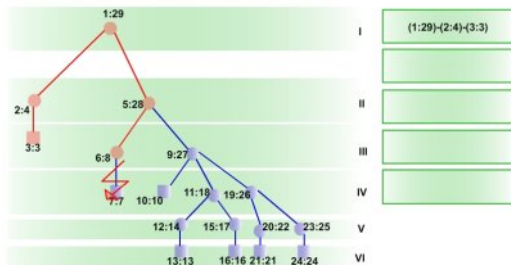
Beispiel(PathStack/6)

report[title='unterwegs']/section[title='Jacken%']



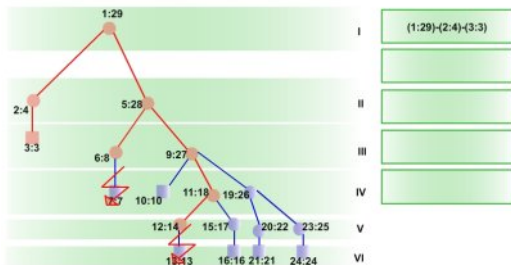
Beispiel(PathStack/7)

report[title='unterwegs']/section[title='Jacken%']



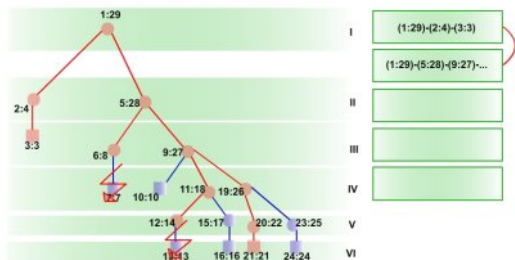
Beispiel(PathStack/8)

`report[title='unterwegs']/section[title='Jacken%']`



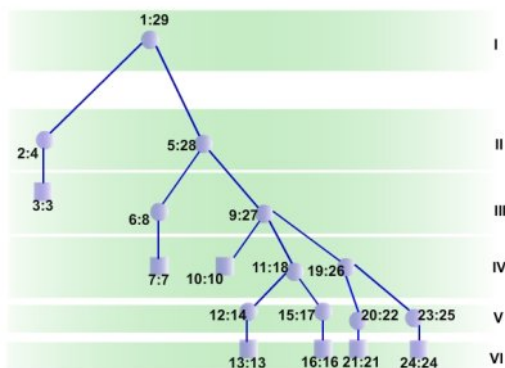
Beispiel(PathStack/9)

report[title='unterwegs']/section[title='Jacken%']



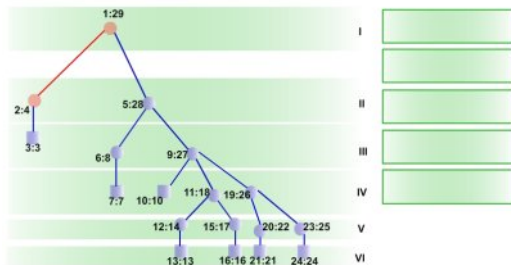
Beispiel(TwigStack/1)

`report[title='unterwegs']/section[title='Jacken%']`



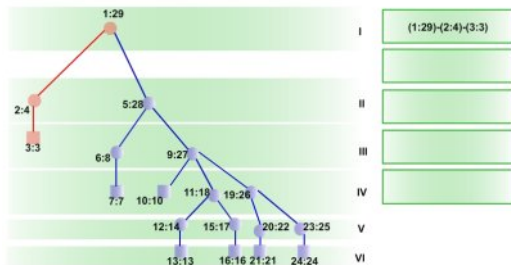
Beispiel(TwigStack/2)

`report[title='unterwegs']/section[title='Jacken%']`



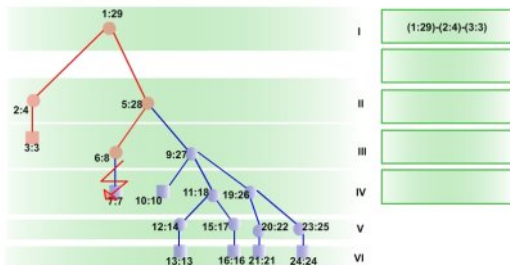
Beispiel(TwigStack/3)

`report[title='unterwegs']/section[title='Jacken%']`



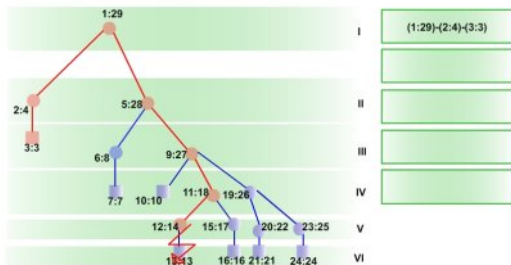
Beispiel(TwigStack/4)

report[title='unterwegs']/section[title='Jacken%']



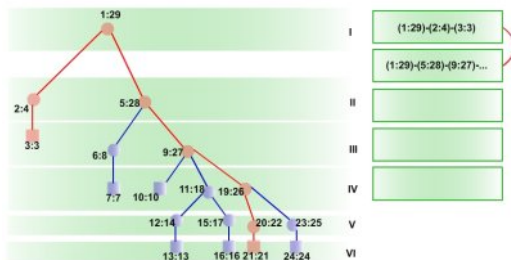
Beispiel(TwigStack/5)

report[title='unterwegs']/section[title='Jacken%']



Beispiel(TwigStack/6)

report[title='unterwegs']/section[title='Jacken%']



PathStack

- Vorteile
 - prüft Zweige von der Wurzel bis zum Blatt
 - hält erfolgreiche und nur teilweise erfolgreiche Anfragepfade im Speicher
- Nachteile
 - ineffizient bei stark einschränkenden Anfragen
 - lange Anfrage-Pfade brauchen Zeit
 - ist somit nicht asytmotisch optimal zur Anfrage

TwigStack

- Vorteile
 - löscht im Speicher gehaltene Teilpfade der Anfrage, wenn sie nicht erfolgreich sind
 - somit Linearität bezüglich Anfrage
- Nachteile
 - lange Anfrage-Pfade brauchen Zeit

TwigStackXB

- Vorteile
 - Sub-Linearität bezüglich der Anfrage
 - beste Ergebnisse bei [4,64] Knoten
 - bei komplexen Anfragen bessere Ergebnisse wie zuvor
- Nachteile
 - viele interne Zwischenergebnisse notwendig (internal pages)
 - einfache Anfragen sind schlechter gestellt