Höhenbalanzierter (perfektbalanzierter) Mehrwegbaum

- Externspeicher-Datenstruktur
  - Eine der häufigsten Datenstrukturen in Datenbanksystemen
- Dynamische Datenstruktur (Einfügen und Löschen)
- Algorithmen für Einfügen und Löschen erhalten die Balanzierungs-eigenschaft
- Garantiert einen begrenzten (worst-case) Aufwand für Zugriff, Einfügen und Löschen
- Der Aufwand für die Operationen Zugriff, Einfügen und Löschen ist bedingt durch die Baumstruktur maximal von der Ordnung log n (O(log n))
- Besteht aus Index und Daten
- Der Weg zu allen Daten ist gleich lang

Definition B+-Baum der Ordnung k

Alle Blattknoten haben die gleiche Tiefe (gleiche Weglänge zur Wurzel)

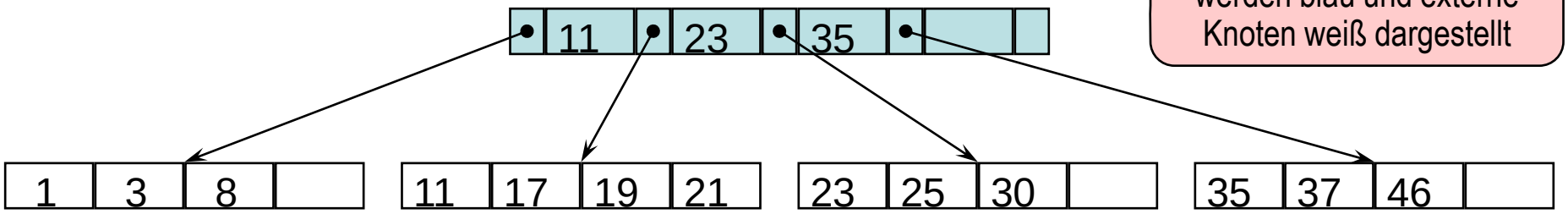
Die Wurzel ist entweder ein Blatt oder hat mind. 2 und max. 2k+1 Kinder

Jeder (interne) Knoten besitzt mindestens k und maximal 2k Schlüsselwerte, daher mindestens k+1 und maximal 2k+1 Kinder

- Für jeden Indexeintrag gilt, dass im linken Unterbaum nur Werte gespeichert sind, die kleiner sind als der Indexeintrag und im rechten Unterbaum nur Werte, die größer oder gleich dem Indexeintrag sind.
- Intervallbasierte Suchdatenstruktur

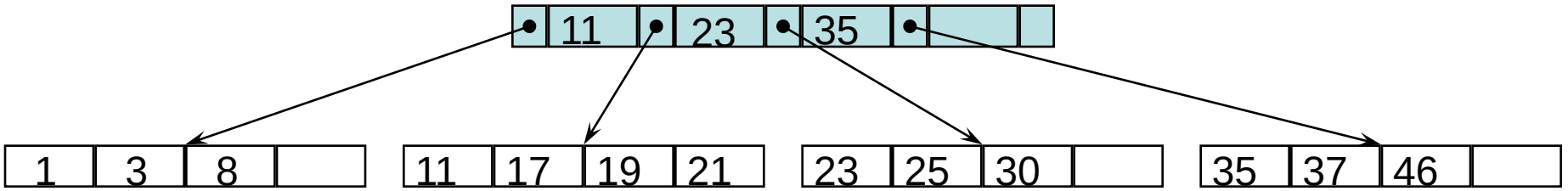
Beispiel: B+-Baum der Ordnung 2

Anmerkung: Indexknoten werden blau und externe Knoten weiß dargestellt



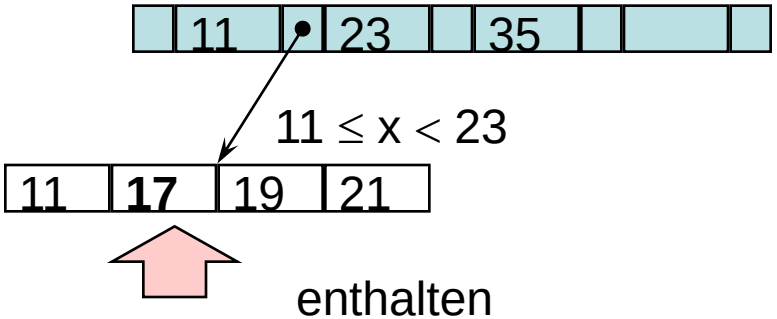
- Jeder Indexknoten hat mindestens 2 und maximal 4 Schlüsselwerte und folglich mindestens 3 und maximal 5 Kinder
- Ausnahme ist die Wurzel, kann auch nur 1 Schlüsselwert mit 2 Kindern enthalten
- Die Größe der externen Knoten ist eigentlich durch die Ordnung nicht definiert, wird aber üblicherweise in der Literatur gleichgesetzt

Suche



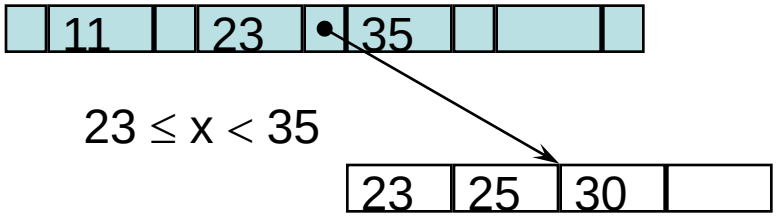
Suchen

Element 17



enthalten

Element 27



nicht enthalten ?

Aufwand der Suche

Höhe eine B^+-Baumes der Ordnung k mit *Datenblockgröße* b (mind. b, max 2b Elemente) ist maximal $\log_{k+1}(n/b)$.

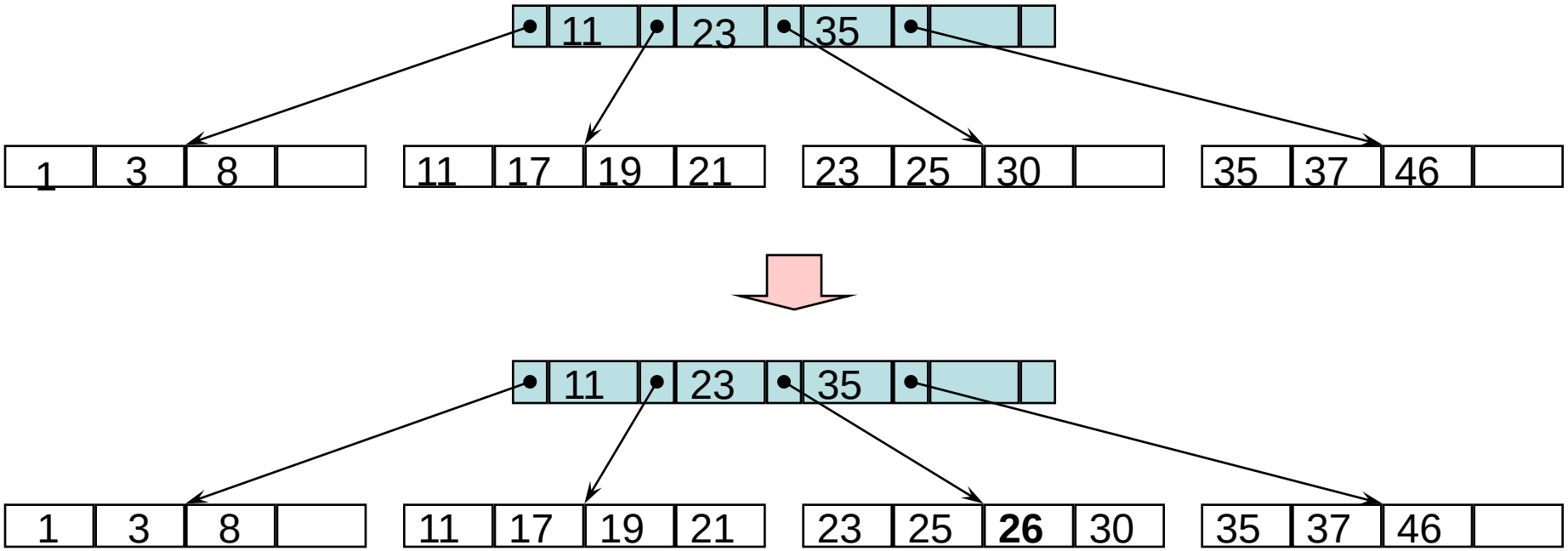
Bereichsabfrage möglich

Suche alle Element im Bereich [Untergrenze, Obergrenze]

B^+-Baum Einfügen (1)

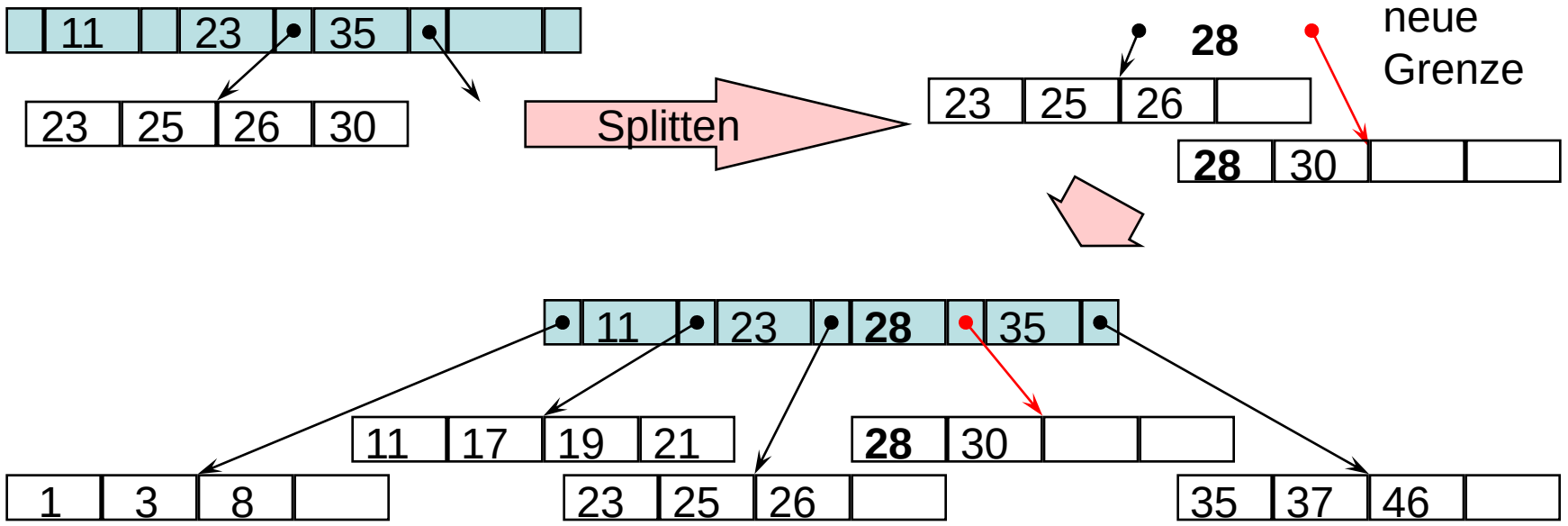
Fall 1: Einfügen Element 26

Platz im externen Knoten vorhanden, einfaches Einfügen



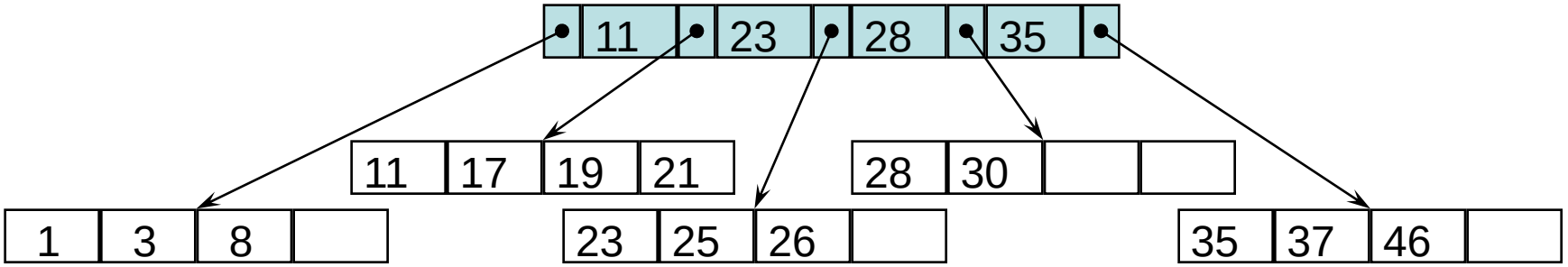
Fall 2: Einfügen Element 28

Kein Platz mehr im externen Knoten (*Überlauf, Overflow*), externer Knoten muss geteilt werden ⇒ Split



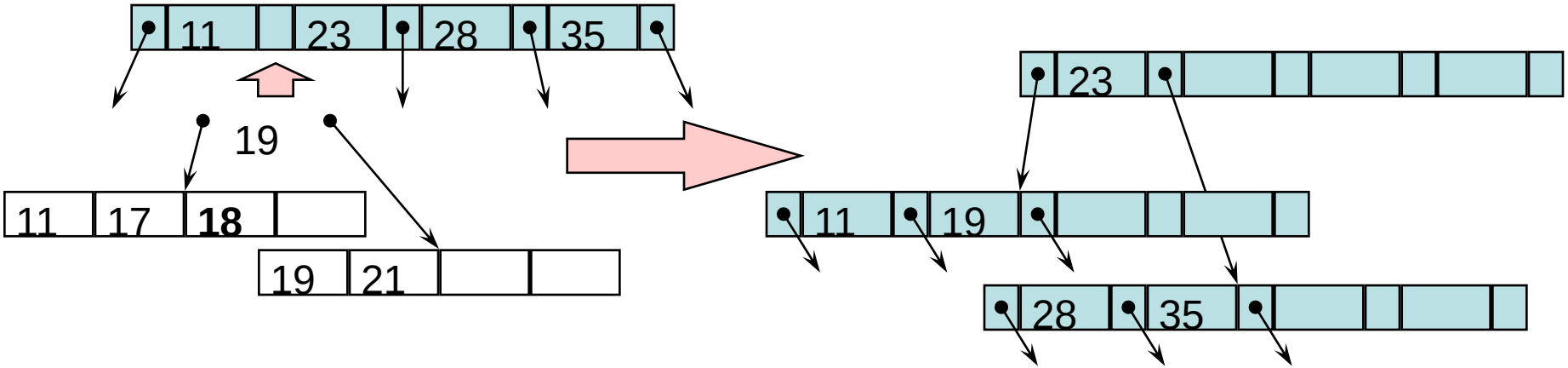
Fall 3: Einfügen Element 18

Kein Platz mehr im Knoten und kein Platz mehr im darüberliegenden Indexknoten, Indexknoten muss gesplittet werden



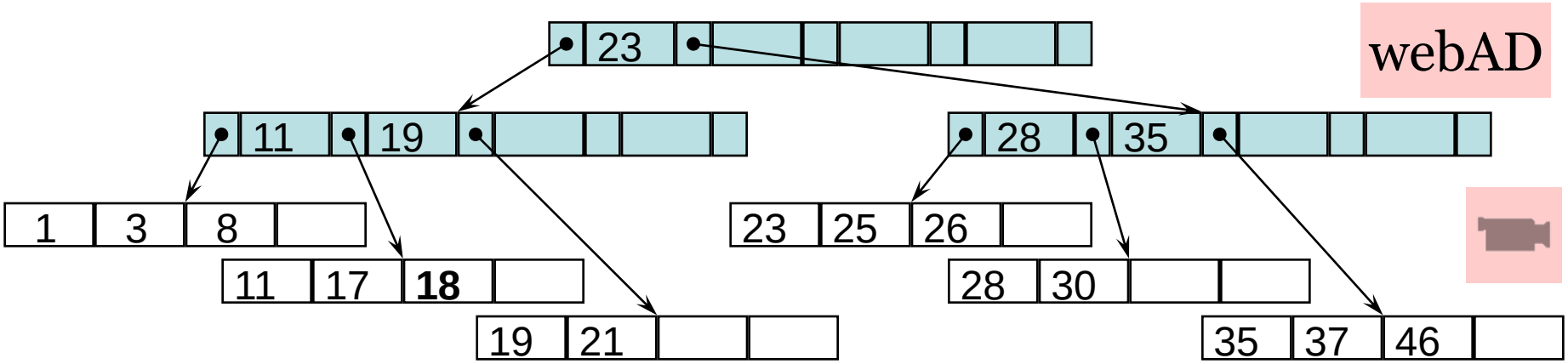
B^+-Baum Einfügen (4)

Beim Splitten eines Indexknoten wird das mittlere Indexelement im darüberliegenden Indexknoten eingetragen. Falls der nicht existiert (*Wurzelsplit*), wird eine neue Wurzel erzeugt



B^+-Baum Einfügen (5)

Der resultierende Baum hat folgendes Aussehen



Unterscheidung des Splits für externe und interne Knoten

Externe Knoten: 2k+1 Elemente werden auf 2 benachbarte externe Knoten aufgeteilt

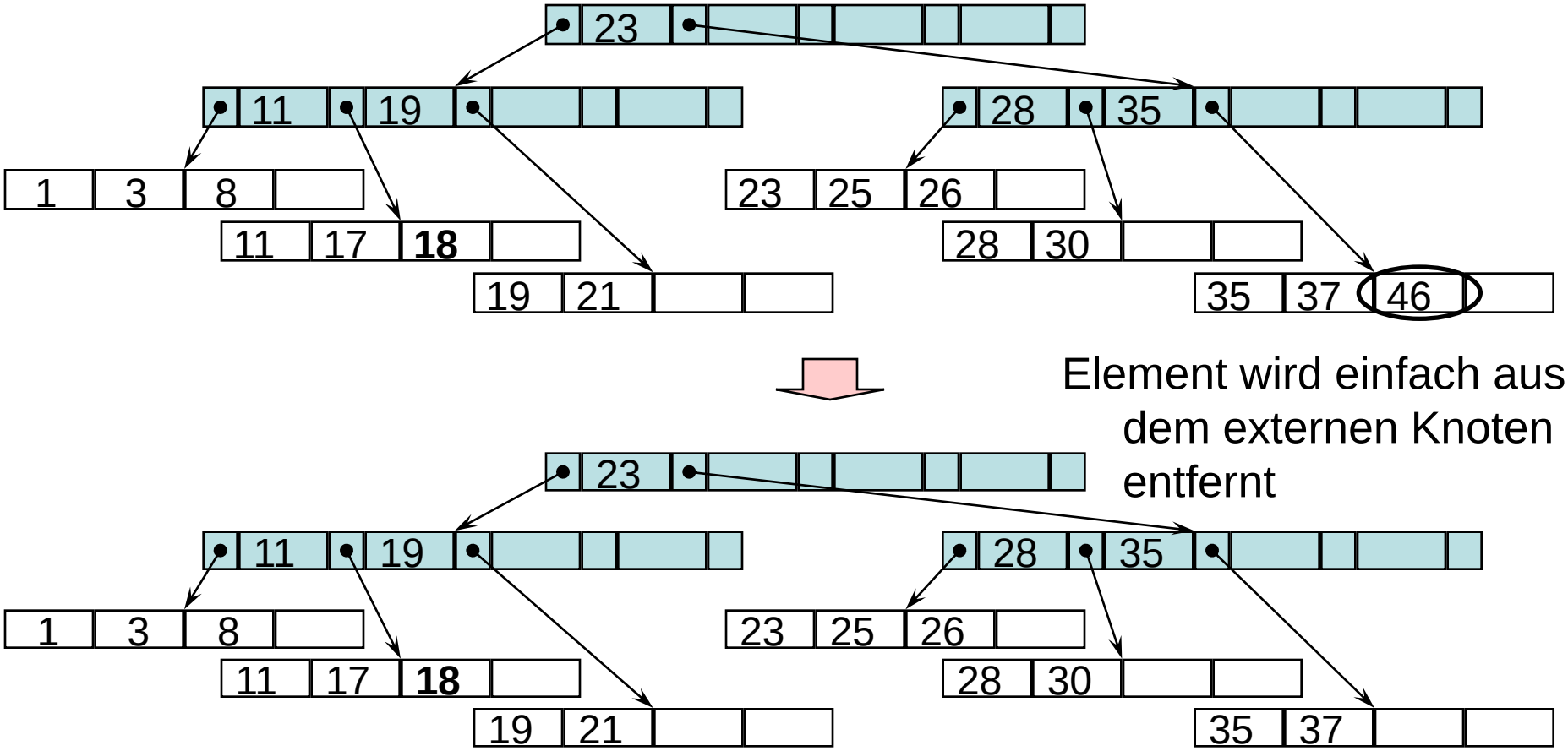
Interne Knoten: 2k+1 Indexelemente (Schlüsselwerte) werden auf 2 benachbarte Indexknoten zu je k Elemente aufgeteilt, das mittlere Indexelement wird in die darüberliegende Indexebene eingetragen.

Falls keine darüberliegende Ebene existiert, wird eine neue Wurzel erzeugt, der Baum wächst um eine Ebene (“Baum wächst von den Blättern zur Wurzel”), der Zugriffsweg zu den Daten erhöht sich um 1.

B^+-Baum Löschen (1)

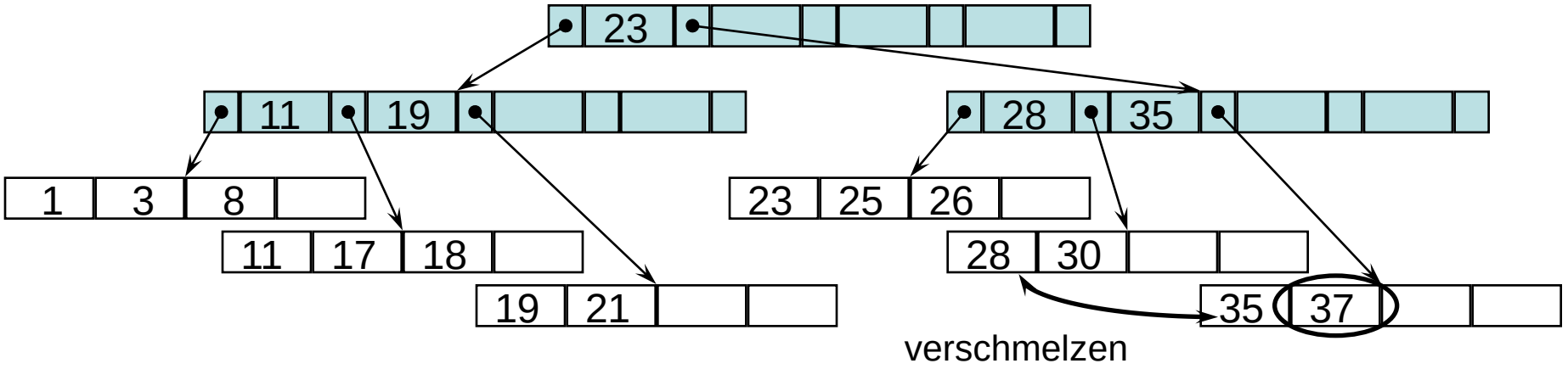
Fall 1: Entfernen von Element 46

Externer Knoten nach Löschen nicht unterbesetzt



Fall 2: Entfernen von Element 37

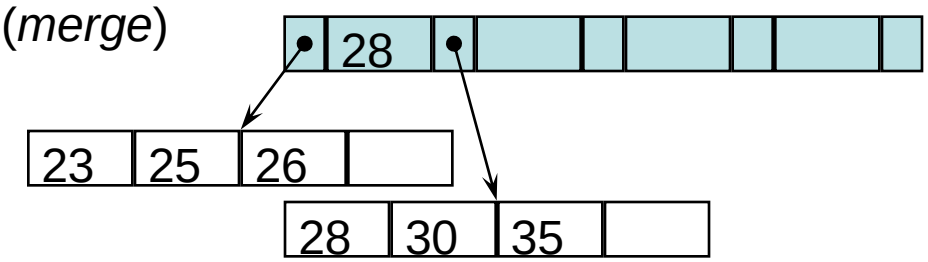
Externer Knoten nach dem Löschen unterbesetzt



Element wird entfernt, ist Knoten unterbesetzt (*Underflow*), Elementanzahl < k

Umgekehrter Vorgang zum Split ⇒ benachbarte Knoten werden *verschmolzen*

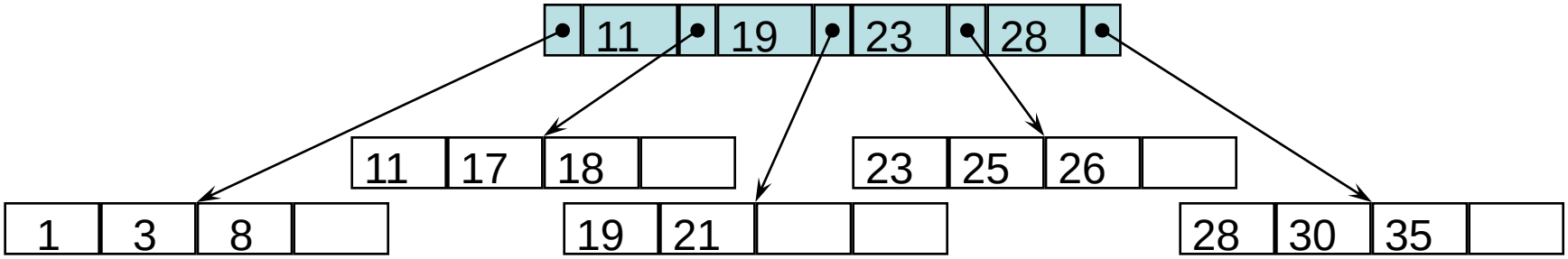
(*merge*)



Interner Knoten unterbesetzt, muss ebenfalls mit Nachbarn verschmolzen werden.

B+-Baum Löschen (3)

Das Verschmelzen kann zur Verringerung der Baumhöhe führen,
2 Knoten werden zur neuen Wurzel verschmolzen, alte Wurzel wird entfernt.

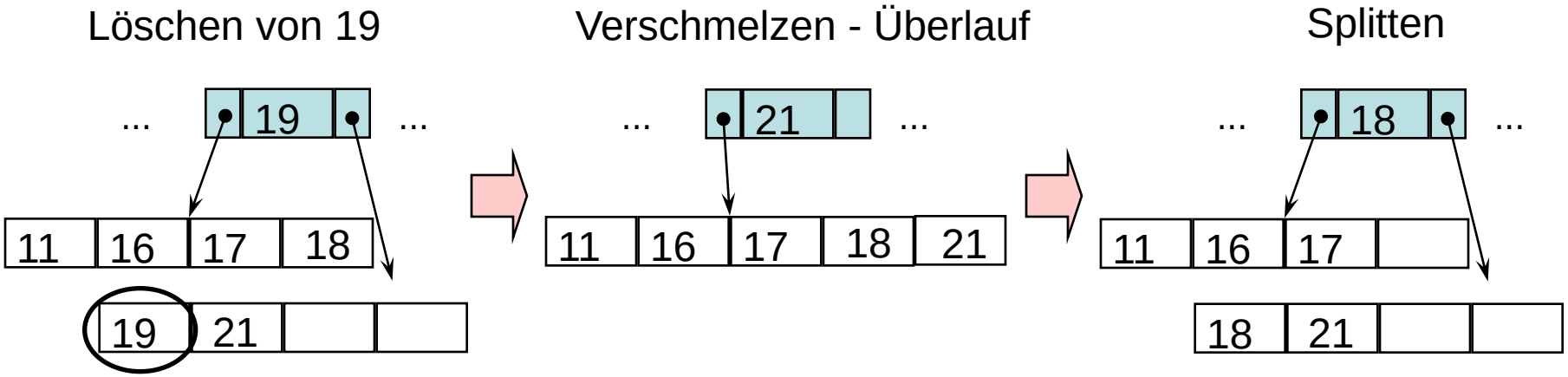


B^+-Baum Löschen (4)

Anmerkung

Das Verschmelzen zweier Knoten kann zu einem Überlauf (analog Einfügen) des neuen Knoten führen, was einen nachfolgenden Split notwendig macht.

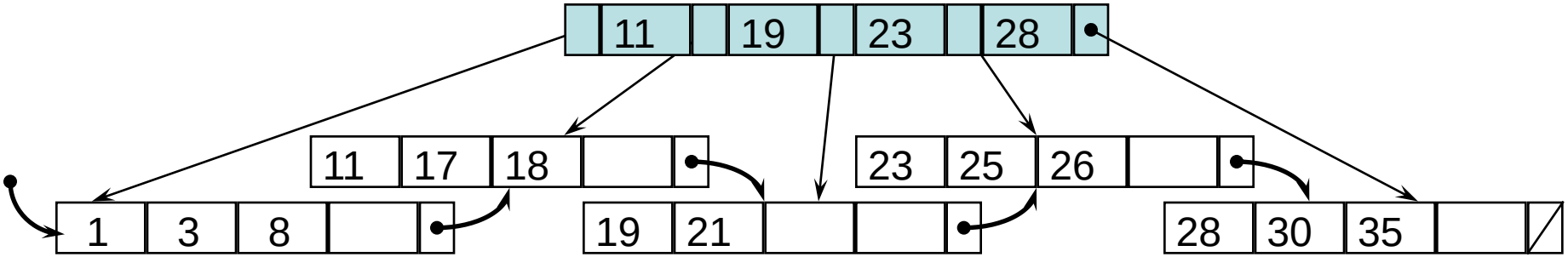
Beispiel: (Baumausschnitt)



Diese (Verschmelzen-Splitten) Sequenz erzeugt eine besseren Aufteilung der Datensätze zwischen benachbarten Knoten. Dies wird auch hier (siehe 2-3-4 Baum, eigentlich fälschlicherweise) als Datensatzverschiebung (shift) bezeichnet.

Datenblockverkettung

In der Praxis werden die Datenblöcke linear verkettet, um einen effizienten, sequentiellen Zugriff in der Sortierreihenfolge der gespeicherten Elemente zu ermöglichen



Analyse B+ - Baum

- Datenverwaltung
  - Einfügen und Löschen unterstützt
- Datenmenge
  - unbeschränkt
    - abhängig von der Größe des vorhandenen Speicherplatzes
- Modelle
  - Externspeicherorientiert
  - Unterstützung komplexer Operationen
    - Bereichsabfragen, Sortierreihenfolge

Analyse B+ - Baum (2)

universität wien

Speicherplatz	O(n)
Split, Verschmelzen	O(1)
Zugriff	O(log n)
Einfügen	O(log n)
Löschen	O(log n)
Sortierreihenfolge	O(n)

Bitte beachten: Ordnung des Baumes ist konstanter Faktor!