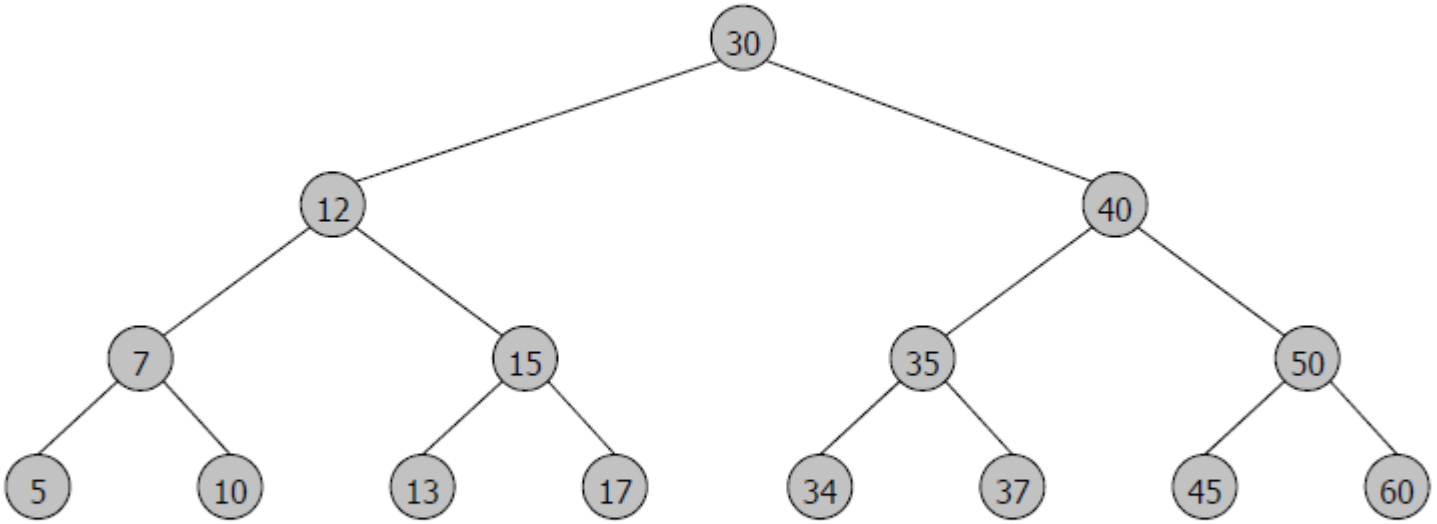
B-drzewa: idea

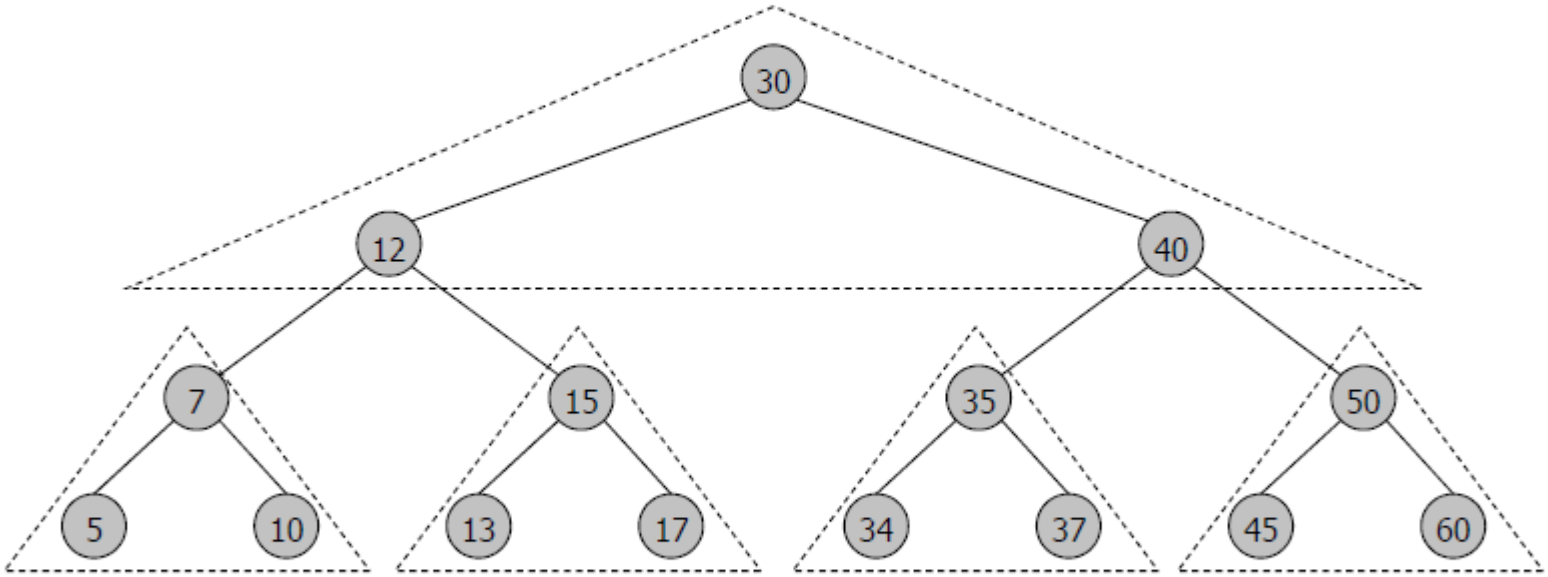
Drzewo decyzyjne, przeszukiwania binarnego:
F = {5, 7, 10, 12, 13, 15, 17, 30, 34, 35, 37, 40, 45, 50, 60}



$$\lceil \log_2(N+1) \rceil \le h \le 1 + \lceil \log_2 N \rceil$$

SDIZO-B drzewo

B-drzewa: idea

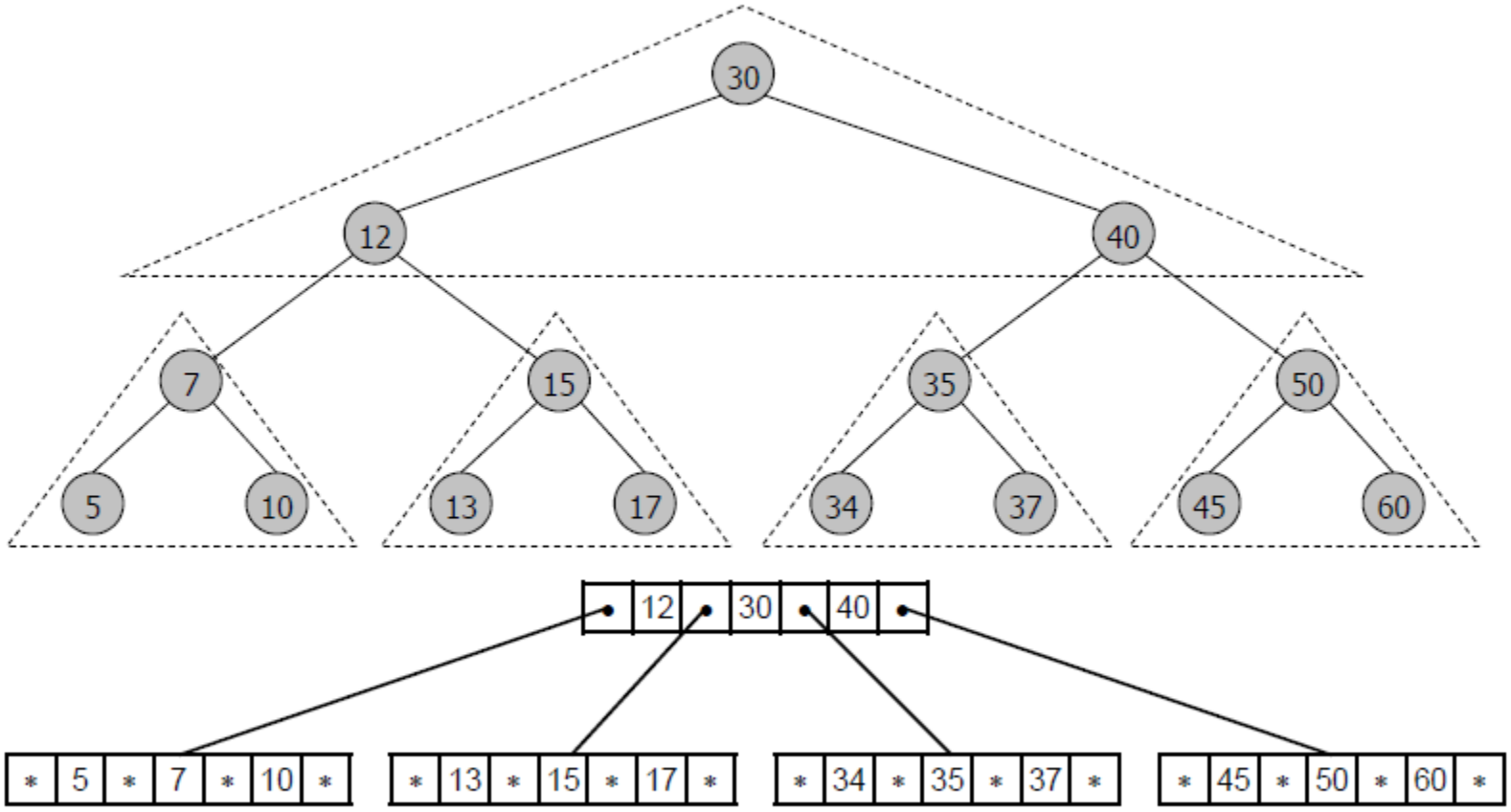


Sposób grupowania wierzchołków uzasadniający postać B-drzewa

T. Pankowski

SDIZO-B drzewo

B-drzewa: idea



Drzewo wielodrogowe utworzone w wyniku pogrupowania drzewa w bloki trójelementowe

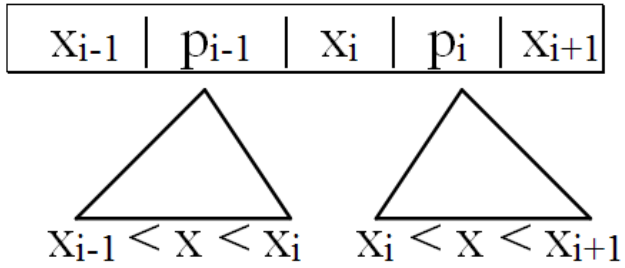
SDIZO-B drzewo

Definicja B-drzewa (perfectly balanced, multiway tree)
Drzewo skierowane T nazywamy B-drzewem klasy t(h, m),
jeśli h = 0 (drzewo puste)
lub

- Wszystkie drogi od korzenia do liści są długości h,
- Każdy wierzchołek z wyjątkiem korzenia ma, co najmniej m kluczy (elementów) i m+1 synów,
- Każdy wierzchołek ma, co najwyżej 2m kluczy (i 2m+1 synów),
- Korzeń ma, co najmniej jeden klucz.

$$p_0 \mid x_1 \mid a_1 \mid p_1 \mid x_2 \mid a_2 \mid p_2 \mid \ldots \mid x_k \mid a_k \mid p_k \mid \ldots$$

x_i – wartość klucza
a_i – adres rekordu w bazie danych posiadający klucz x_i
p_i – wskaźnik na potomka



SDIZO-B drzewo

Inna definicja (dotyczy liczby m) - Cormen

- Każdy wierzchołek z wyjątkiem korzenia ma, co najmniej **m-1** kluczy (elementów) i **m** synów,
- Każdy wierzchołek ma, co najwyżej **2m-1** kluczy (i **2m** synów),

WYSZUKIWANIE

Algorytm zbliżony do wyszukiwanie w drzewie BST, z tym ,że w każdym węźle mamy więcej kluczy od przejrzenia niż jeden. Można zastosować metodę przeszukiwania połówkowego lub liniową.

procedura SZUKAJ (x) – zwraca adres wierzchołka zawierającego klucz x lub adres wierzchołka do którego należy ten klucz wstawić

WSTAWIANIE KLUCZA DO B-DRZEWA

Chcemy dołączyć element indeksu o kluczu x, tak aby nie naruszyć struktury B-drzewa. Dołączanie poprzedzone jest procedurą SZUKAJ, w wyniku której, albo znajdziemy wierzchołek zawierający klucz x (koniec) albo znajdziemy adres wierzchołka (liścia) do którego należy dołączyć klucz x. Jeśli wierzchołek ma mniej niż 2m elementów to dołączamy nowy klucz. Jeśli wierzchołek ma 2m elementów to następuje tzw. kolizja (przepełnienie, nadmiar), którą rozwiązujemy albo metodą podziału albo metodą kompensacji.

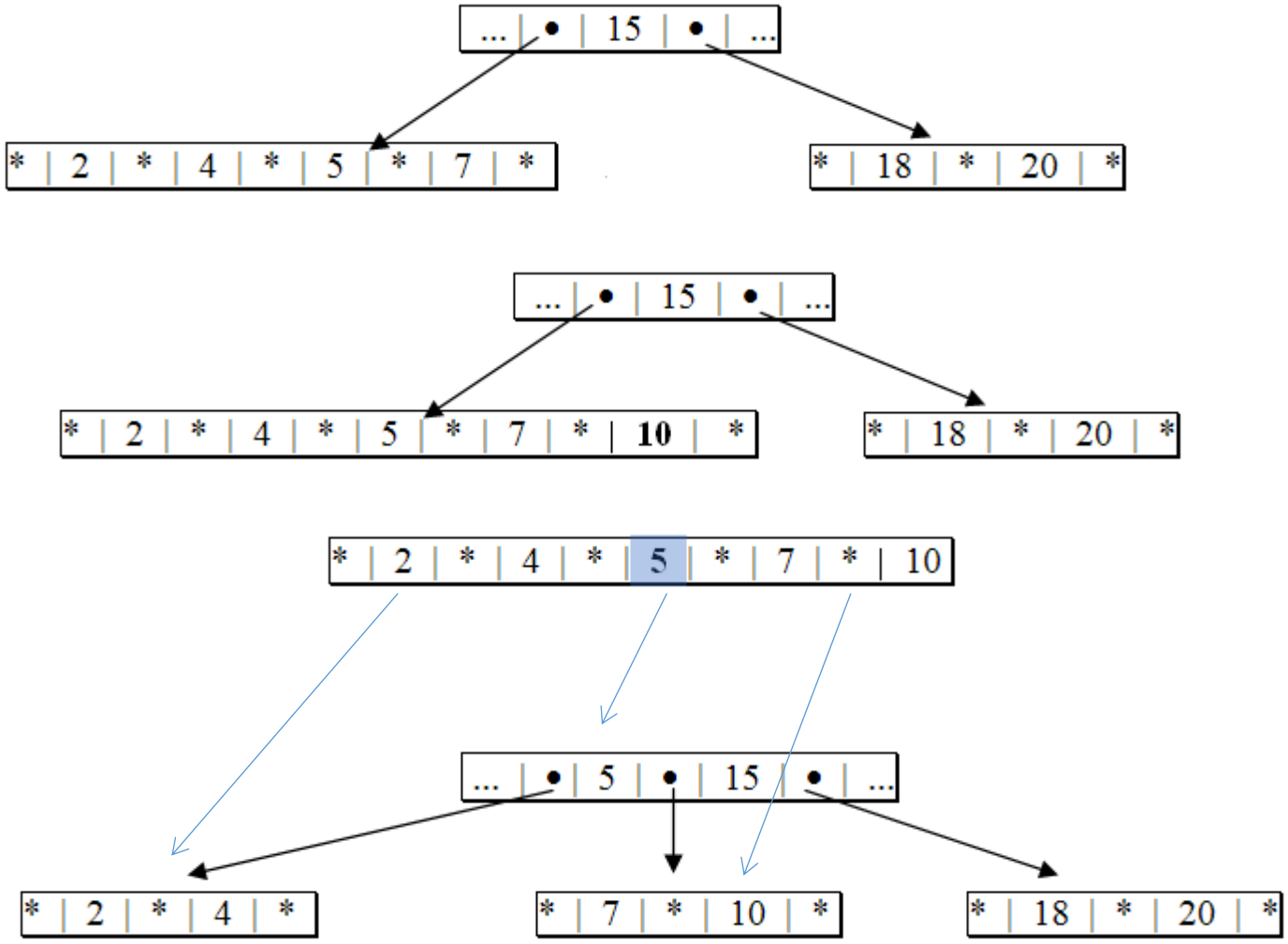
Likwidacja nadmiaru:
a)metoda podziału
b)metoda kompensacji

Metoda podziału :

- **2m+1** elementów dzielimy na trzy części.
- elementy **1, ... , m** umieszczamy w wierzchołku 1
- element **m+1** przenosimy do strony ojca
- elementy **m+2, ... , 2m+1** umieszczamy w wierzchołku 2

SDIZO B-drzewo

m=2, wstawiamy 10



Jeżeli w stronie ojca nastąpi przepełnienie to algorytm powtarzamy

SDIZO B-drzewo

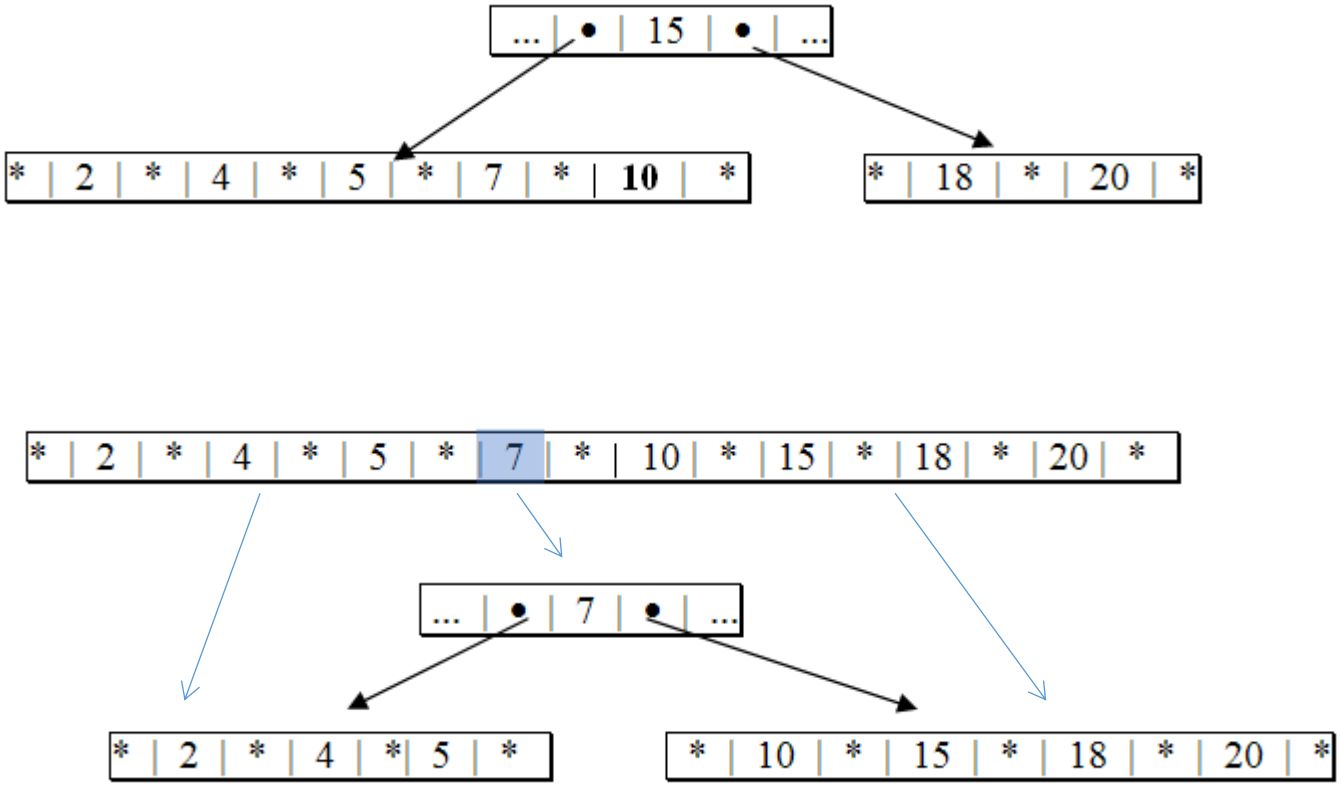
Metoda kompensacji: można ją stosować jeżeli sąsiednia strona zawiera j<2m elementów.

Stan wyjściowy – j=2<2m, obliczamy i=entier((2m+j+3)/2) = ((4+2+3)/2) = 4.

Elementy 1, 2, ..., i-1 umieszczamy w wierzchołku1,

Element i przenosimy do strony ojca,

Elementy i+1, ..., 2m+j+2 umieszczamy w wierzchołku 2.



SDIZO B-drzewo

USUWANIE

- szukamy strony zawierającej klucz x (otrzymamy adres strony)
- jeśli strona jest liściem to usuwamy indeks o kluczu x. Może wówczas wystąpić niedomiar, który usuwamy metodą łączenia lub kompensacji.
- jeżeli strona nie jest liściem to szukamy następnika (poprzednika). Następnik (poprzednik) wstawiany jest w miejsce (x) a następnie usuwany. Może wówczas wystąpić niedomiar, który usuwamy metodą łączenia lub kompensacji.

Likwidacja niedomiaru:

Niech:

j < m – ilość kluczy na stronie z niedomiarem,

k – ilość elementów na sąsiedniej stronie

a)metoda kompensacji (analogicznie jak przy dołączaniu)

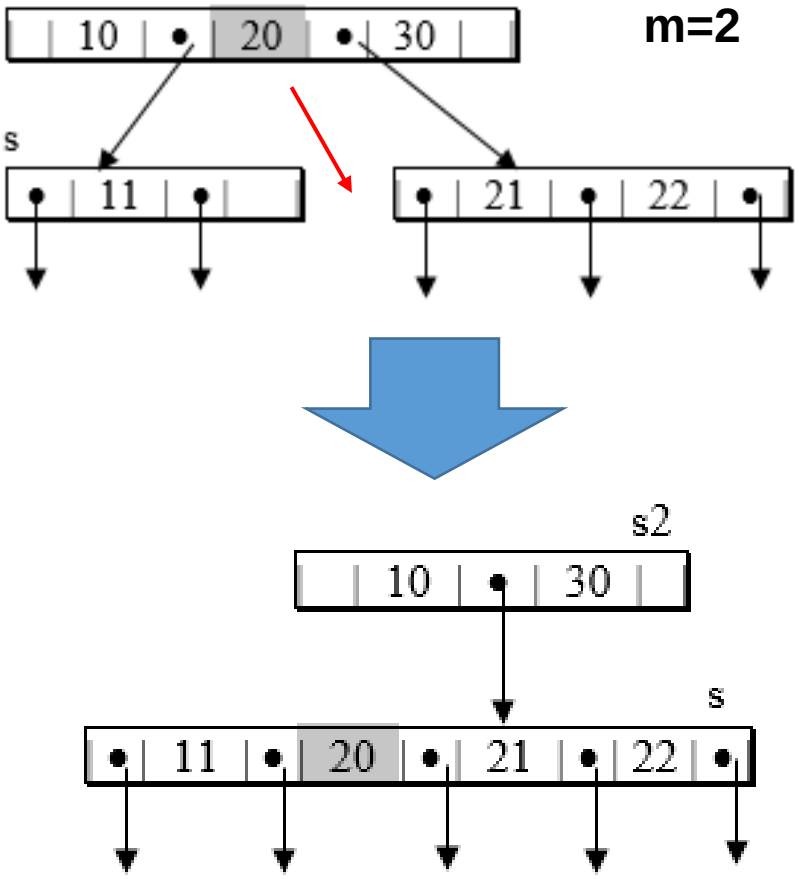
stosujemy jeżeli $j+k \geq 2m$ (bo każda ze stron będzie zwierać co najmniej **m** kluczy spełniając tym warunek B-drzewa)

b) metoda łączenia:

stosujemy jeżeli $j+k < 2m$ (nie da się stworzyć dwóch stron)

SDIZO B-drzewo

Metoda łączenia (stron)



Na stronie s2 może wystąpić niedomiar, trzeba wówczas operację łączenia powtórzyć, lub jeśli s2 jest korzeniem to przekazujemy go do puli stron pustych (drzewo zmniejszy wysokość)

SDIZO B-drzewo

Poziom i	Liczba wierzchołków przy min wypełnieniu W_{min}	Liczba wierzchołków przy max wypełnieniu W_{max}
1	1	1
2	2	$2m+1$
3	$2(m+1)$	$(2m+1)^2$
...	...	...
h	$2(m+1)^{h-2}$	$(2m+1)^{h-1}$
Σ	$1+\frac{2}{m}((m+1)^{h-1}-1)$	$\frac{1}{2m}((2m+1)^h-1)$

Wartości W_{min} i W_{max} można obliczyć stosując wzór na sumę ciągu geometrycznego.

Dla W_{min} na pierwszym poziomie jest jedna strona korzenia. Minimalnie wypełniony korzeń zawiera 1 klucz, czyli dwa wskaźniki na potomków, stąd na drugim poziomie są dwie strony. Na trzecim i pozostałych poziomach liczba stron jest przemnażana przez (m+1) w stosunku do poprzedniego poziomu.
Dla W_{max} korzeń jest maks. wypełniony i zawiera 2m+1 potomków

SDIZO B-drzewo

Obliczanie liczby elementów w B-drzewie:

W_{min} – liczba wierzchołków przy minimalnym wypełnieniu

W_{max} - liczba wierzchołków przy maksymalnym wypełnieniu

N_{min} – liczba kluczy przy minimalnym wypełnieniu

N_{max} - liczba kluczy przy maksymalnym wypełnieniu

$$W_{min} = 1 + 2\sum_{i=1}^{h-1}(m+1)^{i-1} = 1 + \frac{2((m+1)^{h-1}-1)}{m}$$

$$W_{max} = \sum_{i=1}^{h}(2m+1)^{i-1} = \frac{(2m+1)^h - 1}{2m}$$

$N_{min} = 1 + m(W_{min} - 1) = 2(m+1)^{h-1} - 1$

bo korzeń zawiera 1 klucz, a pozostałe węzły(W_{min}-*1*) po *m* kluczy

$N_{max} = 2mW_{max} = (2m+1)^h - 1$

bo jest W_{max} węzłów, a każdy ma po *2m* kluczy

Szacowanie wysokości B-drzewa:

h_{min} – wysokość przy minimalnym wypełnieniu

h_{max} – wysokość przy minimalnym wypełnieniu

$$N_{min} = 2(m+1)^{h-1} - 1 \Rightarrow h_{min} = 1 + \log_{m+1}\left(\frac{N+1}{2}\right)$$

$$N_{max} = (2m+1)^{h} - 1 \Rightarrow h_{max} = log_{2m+1}(N+1)$$

$$h_{max} \leq h \leq h_{min}$$

$$log_{2m+1}(N+1) \leq \boldsymbol{h} \leq 1 + log_{m+1}\left(\frac{N+1}{2}\right)$$

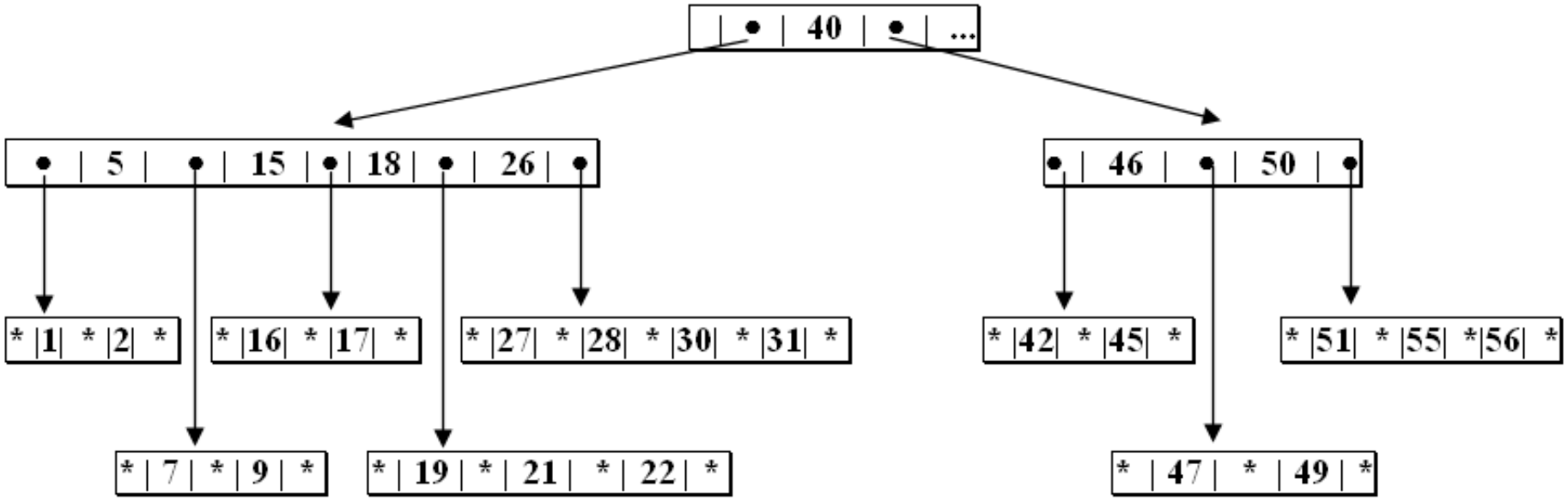
SDIZO B-drzewo

ZADANIE

Dane jest B-drzewo klasy t(h, 2) (jak na rysunku):

a) wstaw obiekt o kluczu 32 (metodą podziału i metodą kompensacji),

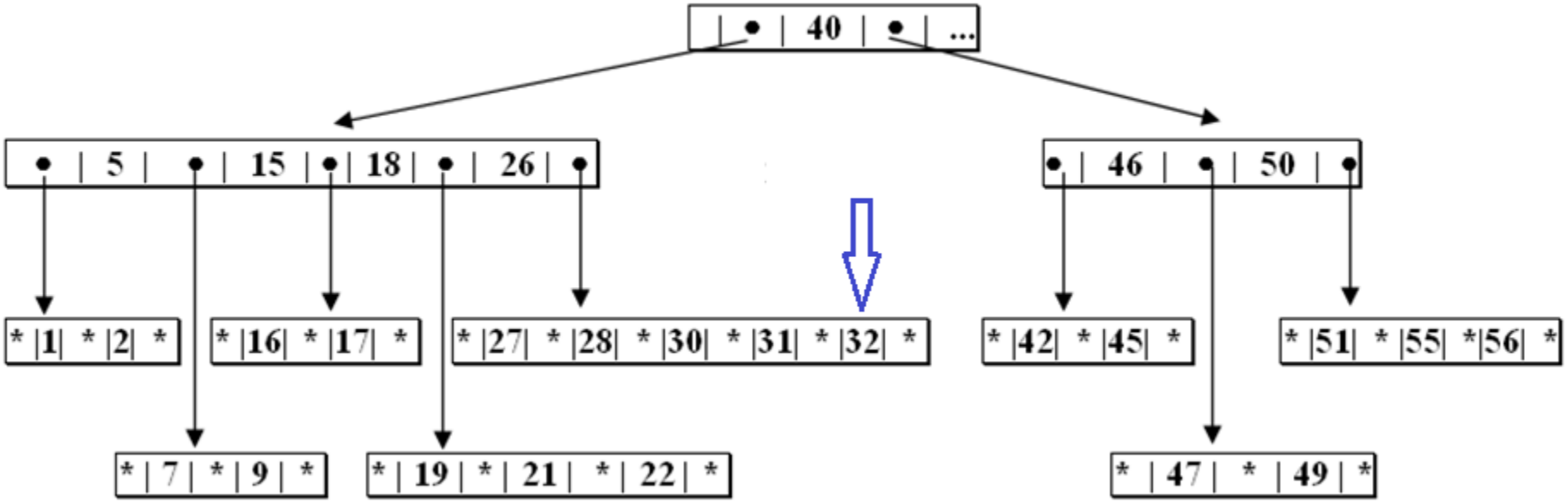
b) usuń obiekt o kluczu 46 (metodą łączenia i metodą kompensacji).



SDIZO B-drzewo

Wstawianie klucza 32 metodą podziału (1)

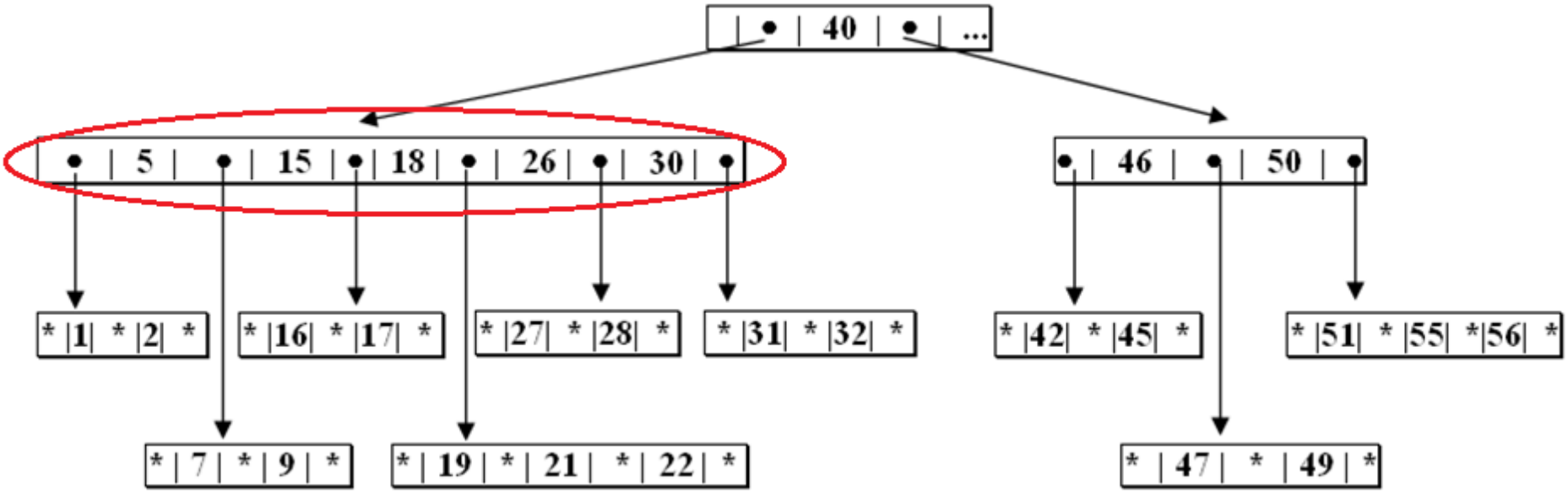
1) Wstawiamy klucz 32 jak na rysunku (następuje przepełnienie – ilość elementów na stronie > 2*m czyli 4)



2) Dzielimy przepełnioną stronę na dwie części względem środkowego elementu

Wstawianie klucza 32 metodą podziału (2)

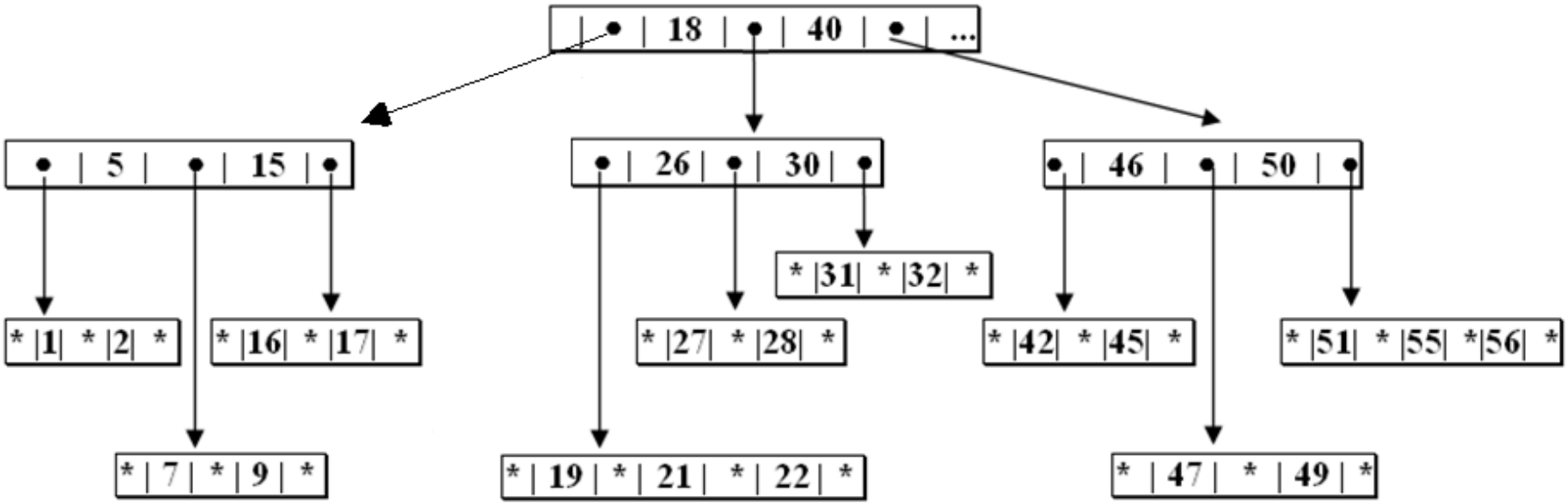
3) Wstawiamy klucz 30 wraz ze stronami na stronę rodzica. W wyniku tej operacji na stronie rodzica nastąpi przepełnienie



4) Dzielimy przepełnioną stronę na dwie części względem środkowego elementu (jak poprzednio)

Wstawianie klucza 32 metodą podziału (3)

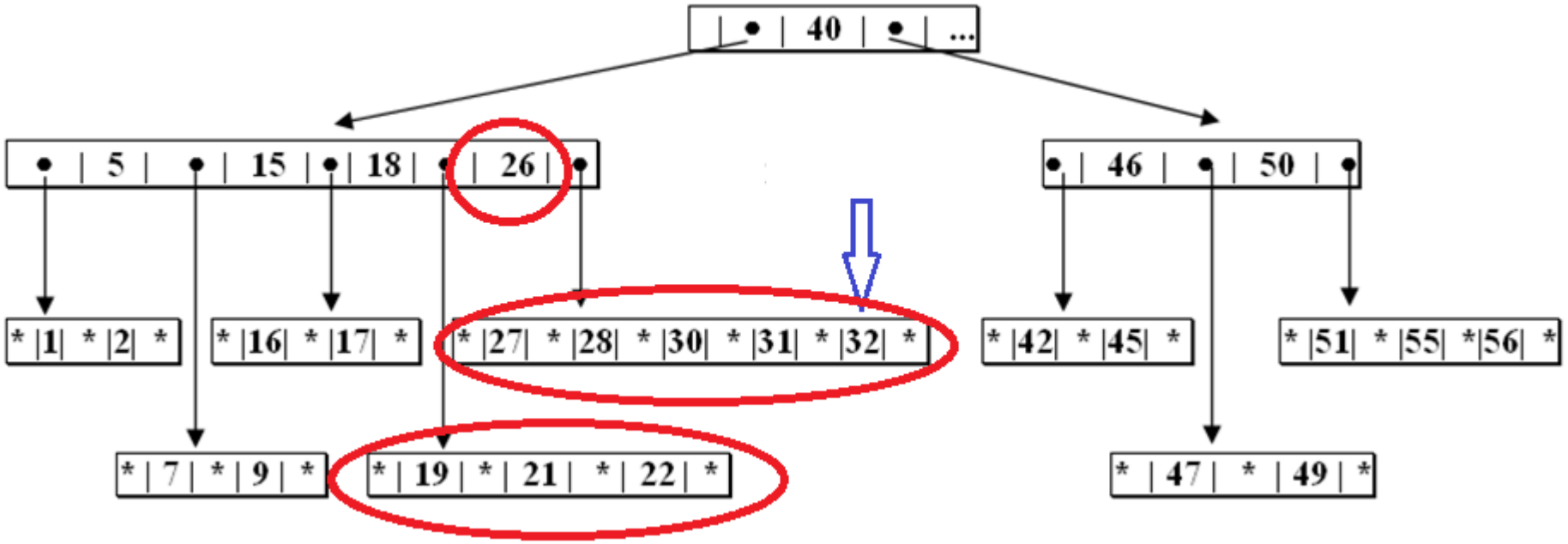
5) Wstawiamy klucz 18 wraz ze stronami na stronę rodzica. W wyniku tych operacji B-drzewo wygląda jak na rysunku niżej.



SDIZO B-drzewo

Wstawianie klucza 32 metodą kompensacji (1)

1) Wstawiamy klucz 32 jak na rysunku (następuje przepełnienie). Likwidujemy przepełnienie poprzez przekopiowanie elementów nadmiarowych z przepełnionej strony na stronę sąsiadującą na której ilość elementów < 2*m. W procesie reorganizacji bierze udział rodzic obu stron.



2) Reorganizacja polega na połączeniu w jeden ciąg zaznaczonych na czerwono elementów, wybranie elementu środowego i zamienieniu go z rodzicem (26)

SDIZO B-drzewo

Wstawianie klucza 32 metodą kompensacji (2)

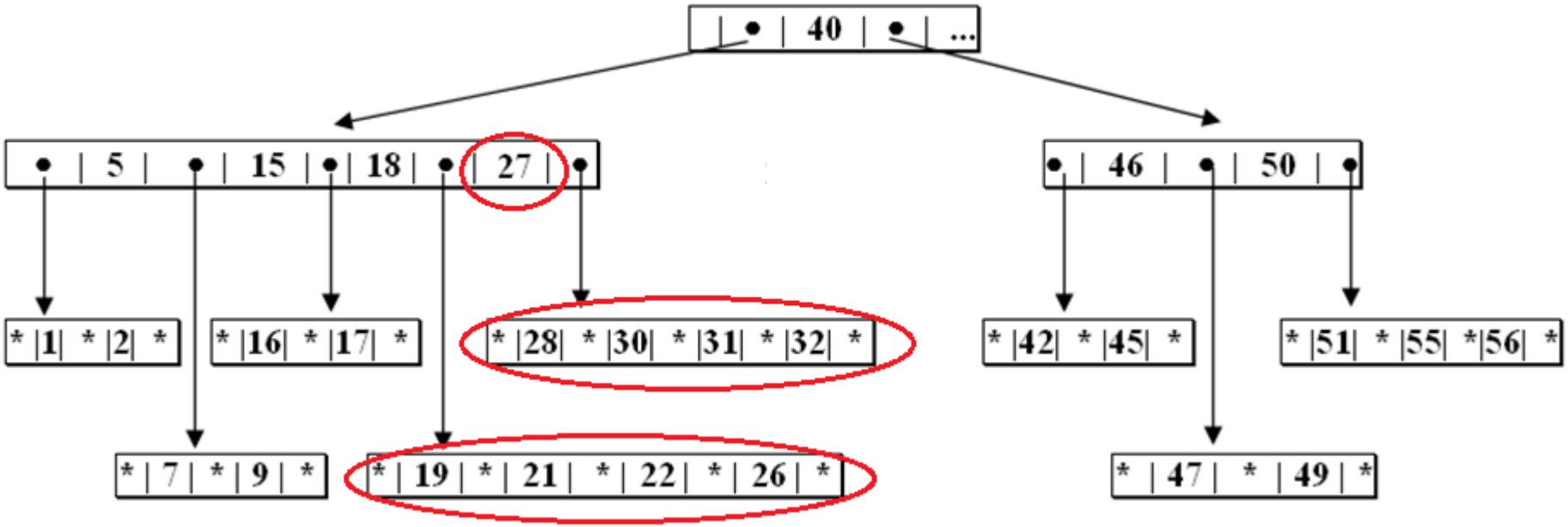
W tym celu:

3) Tworzymy ciąg z zaznaczonych elementów co daje następujący ciąg kluczy:

19, 21, 22, 26, 27, 28, 30, 31, 32

4) Wybieramy środkowy element tj. 27. Dzieli on jednocześnie ciąg na dwa podciągi 19,21,22,26 oraz 28,30,31,32 które będą tworzyć zreorganizowane strony

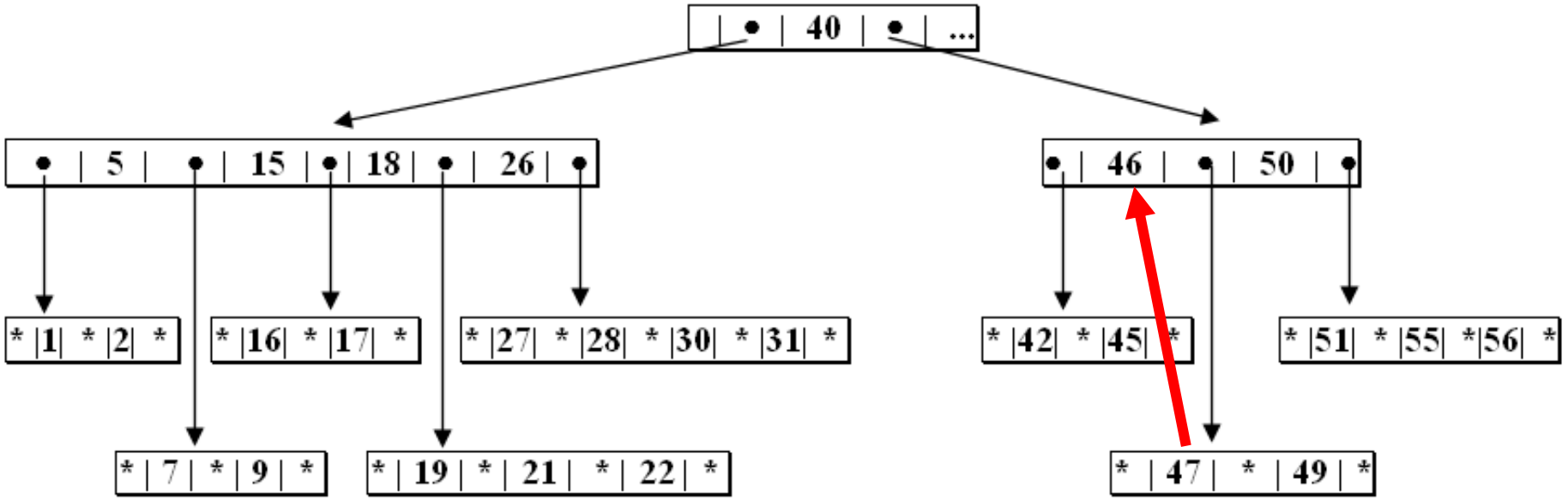
5) Wstawiamy element środkowy (27) w miejsce rodzica (26), a zreorganizowane strony w miejsce poprzednich. Po tych operacjach drzewo wygląda jak na rys. niżej. Kolorem czerwonym zaznaczone zmienione elementy.



SDIZO B-drzewo

USUWANIE elementu 46 z użyciem następnika i metody łączenia (1)

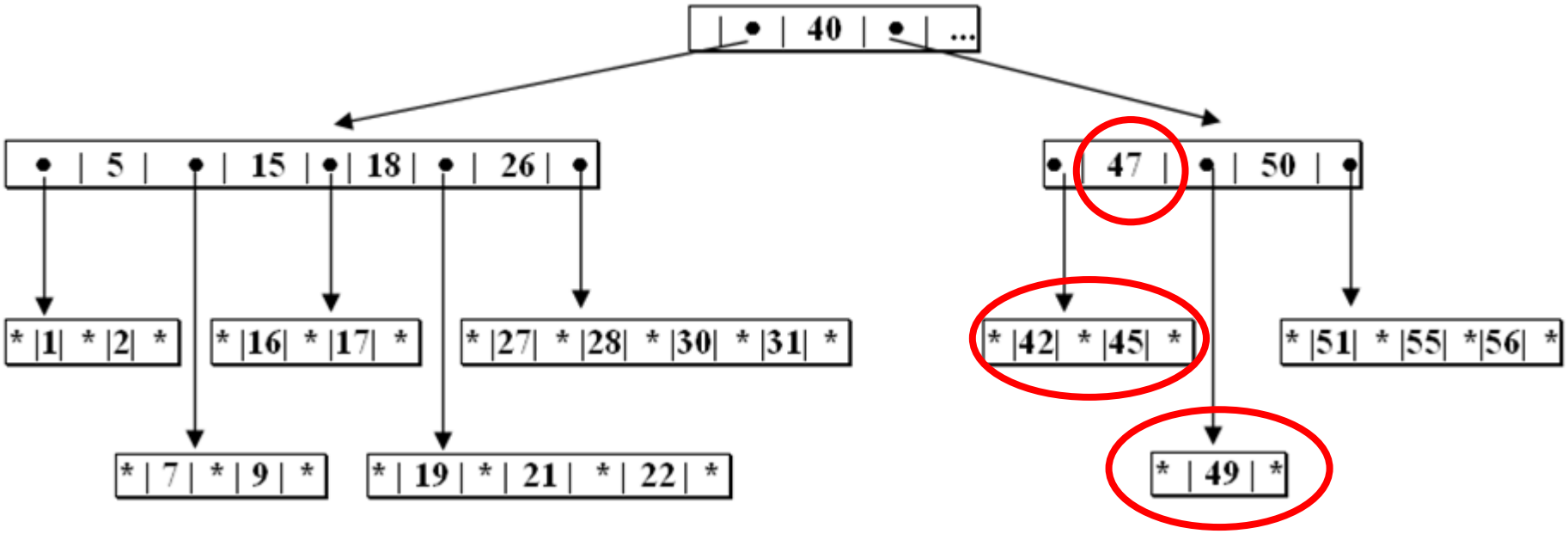
1. Wyszukujemy następnika 46->jest nim element 47 (najmniejszy element w prawym poddrzewie na stronic będącej liściem)
2. Wstawiamy w miejsce usuwanego elementu następnik
3. Usuwamy 47 ze swojej pierwotnej pozycji



SDIZO B-drzewo

USUWANIE elementu 46 z użyciem następnika i metody łączenia (2)

1. Na stronie z elementem 49 pojawia się niedomiar, który likwidujemy poprzez połączenie ze stroną sąsiadującą po lewej tj. łączymy elementy ze strony lewej z elementem nadrzędnym oraz stroną z niedomiarem tworząc jedna stronę

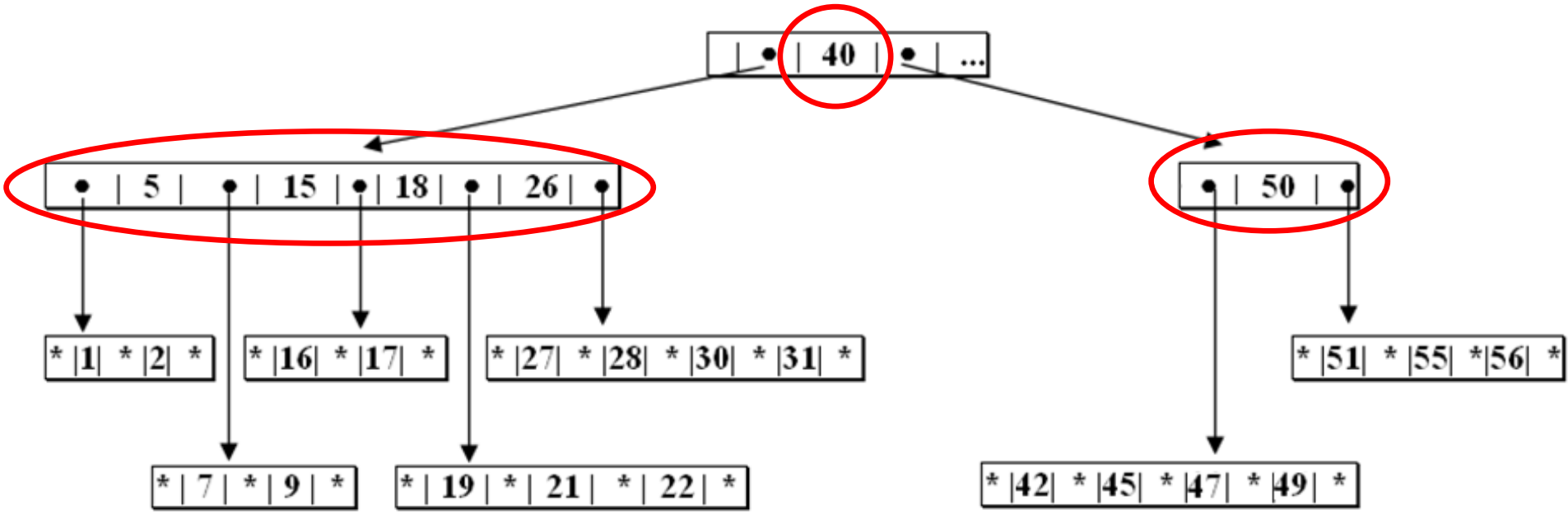


SDIZO B-drzewo

USUWANIE elementu 46 z użyciem następnika i metody łączenia (3)

2. Po przesunięciu elementu 47 na stronę liścia powstanie na tej stroni niedobór, który można zlikwidować stosując **tylko** metodę kompensacji ze stroną sąsiadującą

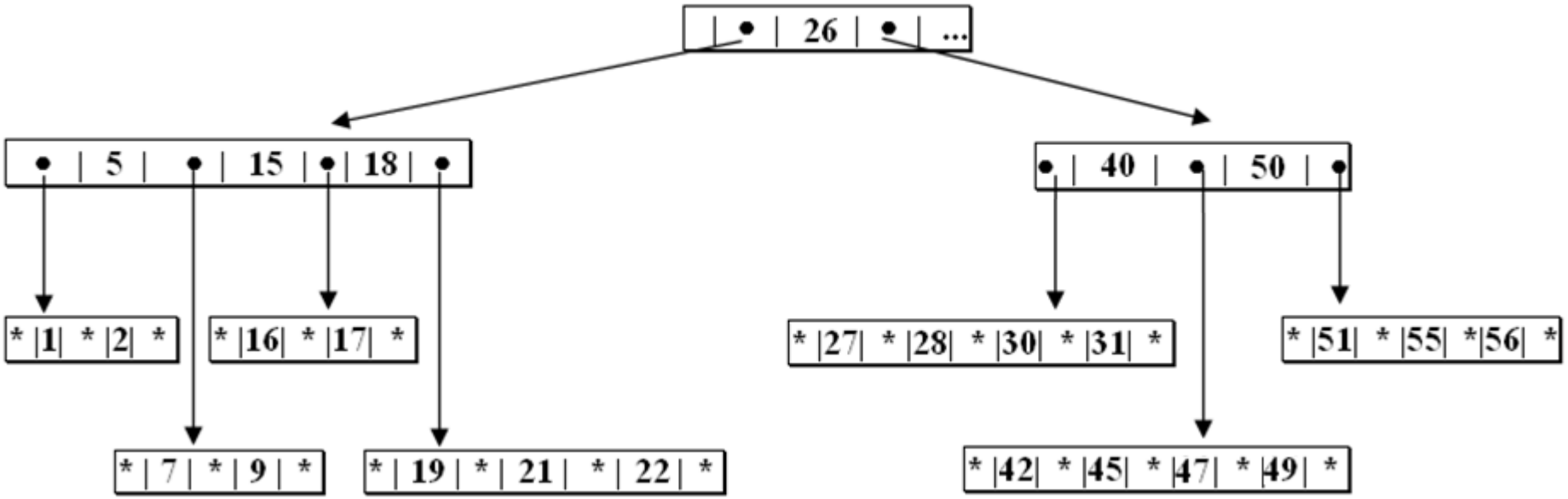
3. Dokonujemy zatem ponownego podziału na strony tj. łączymy elementy z łączonych stron oraz rodzica w jeden ciąg. Wybieramy element środkowy i wymieniamy go ze starym rodzicem (tj. 40), natomiast elementy po lewej i prawej stronie nowego rodzica staja się nowymi stronami (5, 15,18, **26**, 40, 50)



SDIZO B-drzewo

USUWANIE elementu 46 z użyciem następnika i metody łączenia (4)

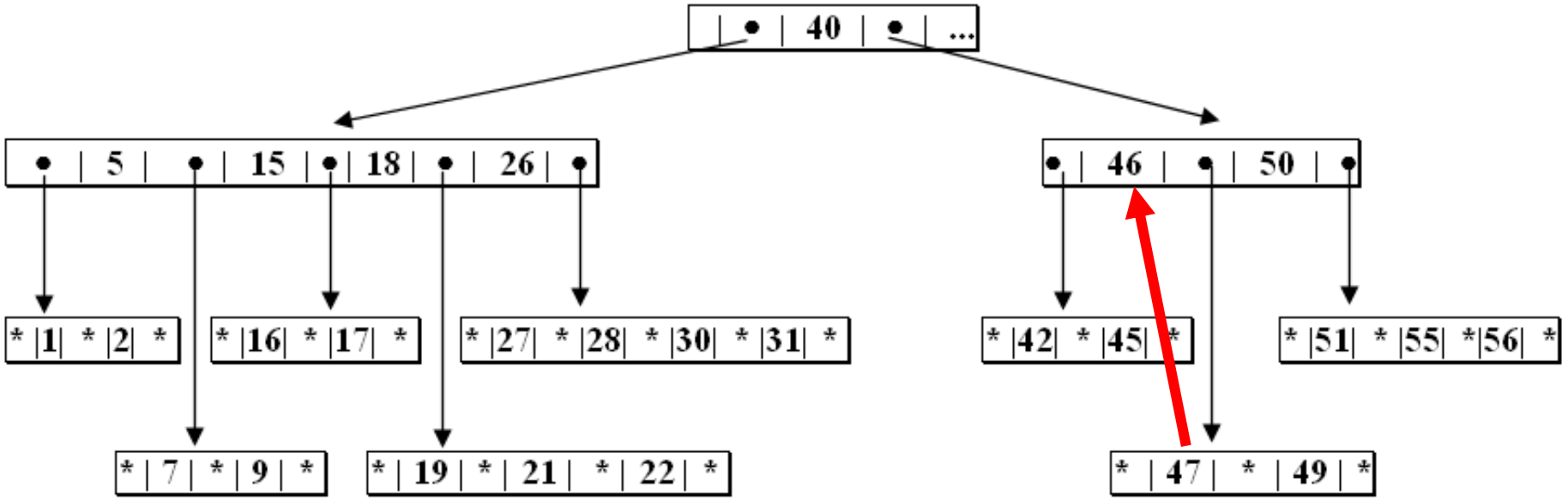
4.Poniżej widok drzewa po całkowitym usunięciu elementu 46 i przywróceniu warunków B-drzewa



SDIZO B-drzewo

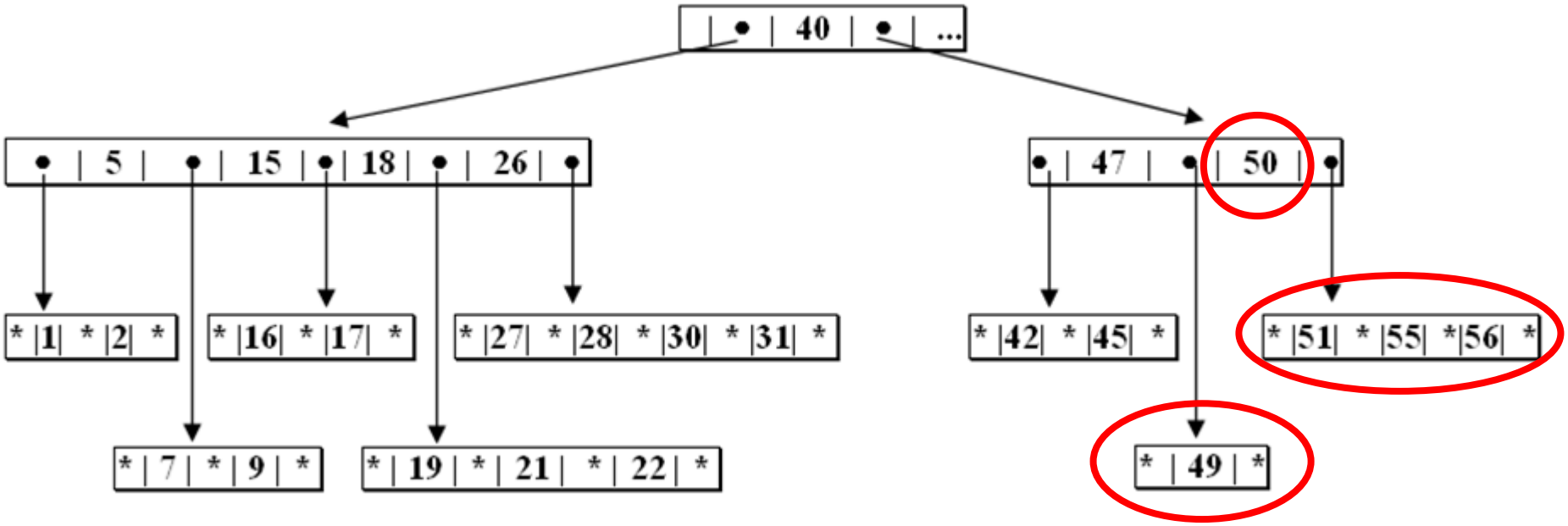
USUWANIE elementu 46 z użyciem następnika i zastosowaniu od razu metody kompensacji (1)

1. Wyszukujemy następnika 46->jest nim element 47 (najmniejszy element w prawym poddrzewie na stronic będącej liściem)
2. Wstawiamy w miejsce usuwanego elementu następnik
3. Usuwamy 47 ze swojej pierwotnej pozycji



USUWANIE elementu 46 z użyciem następnika i zastosowaniu od razu metody kompensacji (2)

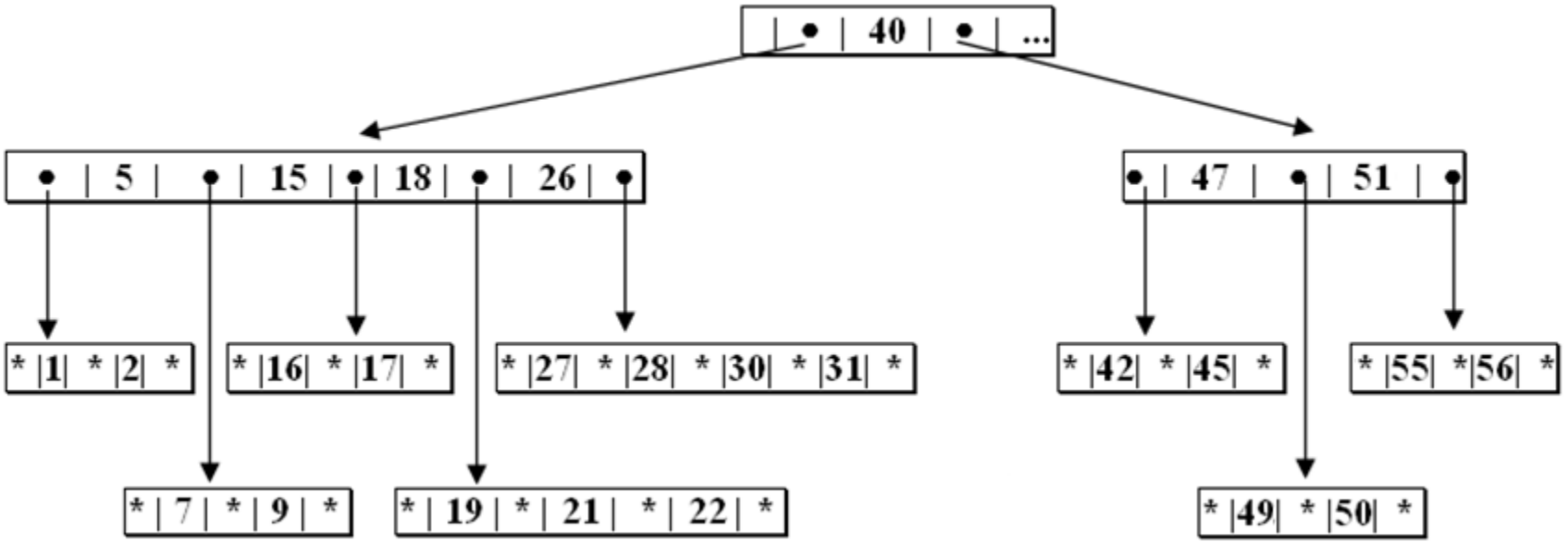
4. Na stronie z elementem 49 pojawia się niedomiar, który likwidujemy metodą kompensacji tj. poprzez połączenie ze stroną sąsiadującą po prawej stronie oraz elementem nadrzędnym (50)



SDIZO B-drzewo

USUWANIE elementu 46 z użyciem następnika i zastosowaniu od razu metody kompensacji (3)

5. Dokonujemy zatem reorganizacji stron. Łączymy 49,50,51,55,56, wybieramy element środkowy (51), zamieniamy go ze starym elementem nadrzędnym, a pozostałe elementy rozmieszczamy odpowiednio po prawej i lewej stronie.



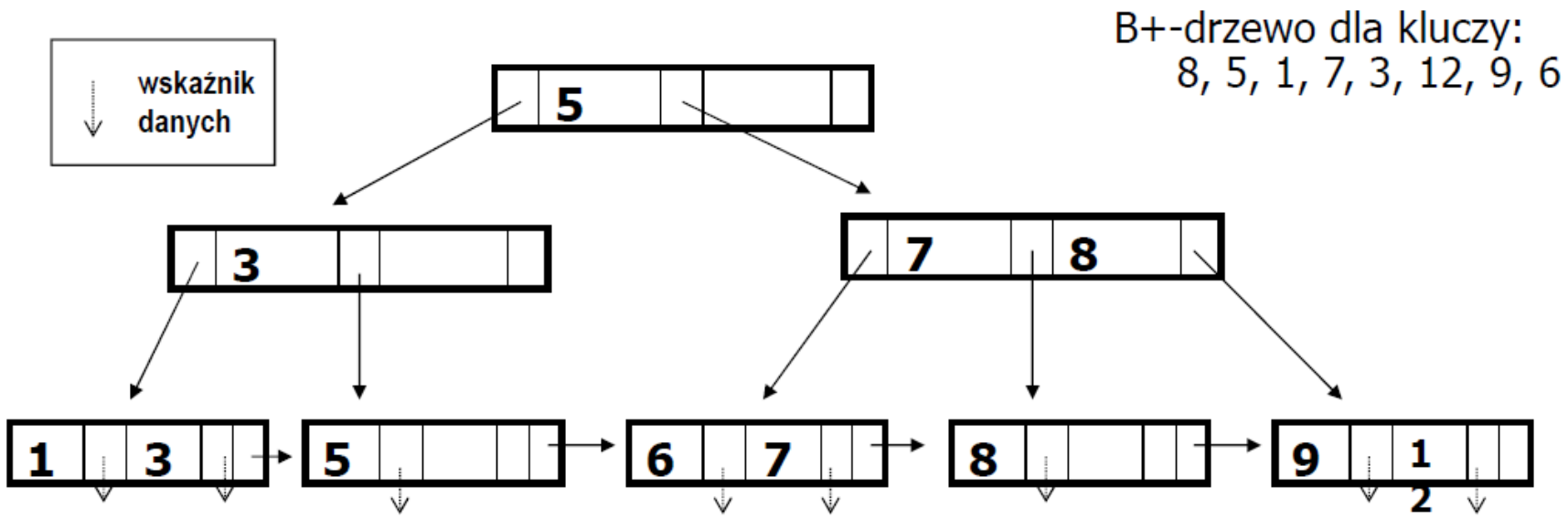
SDIZO B^+ -drzewo

B^+ drzewo

W większości komercyjnych systemów baz danych do tworzenia indeksów wykorzystywana jest pewna modyfikacja B-drzew o nazwie **B^+-drzewa.**

- W przypadku B^+-drzewa wskaźniki danych są przechowywane **tylko w liściach**
- Wierzchołki liści posiadają wpis (indeks ze wskaźnikiem do rekordu danych) dla każdej wartości pola wyszukiwania
- Niektóre wartości pola wyszukiwania są powtarzane w wierzchołkach wewnętrznych i służą do wspomagania wyszukiwania
- Koszt wyszukiwania – **wysokość drzewa + 1.**
- Liście implementowane są dodatkowo jako lista (w celu przyśpieszenia przeszukiwań zakresowych)

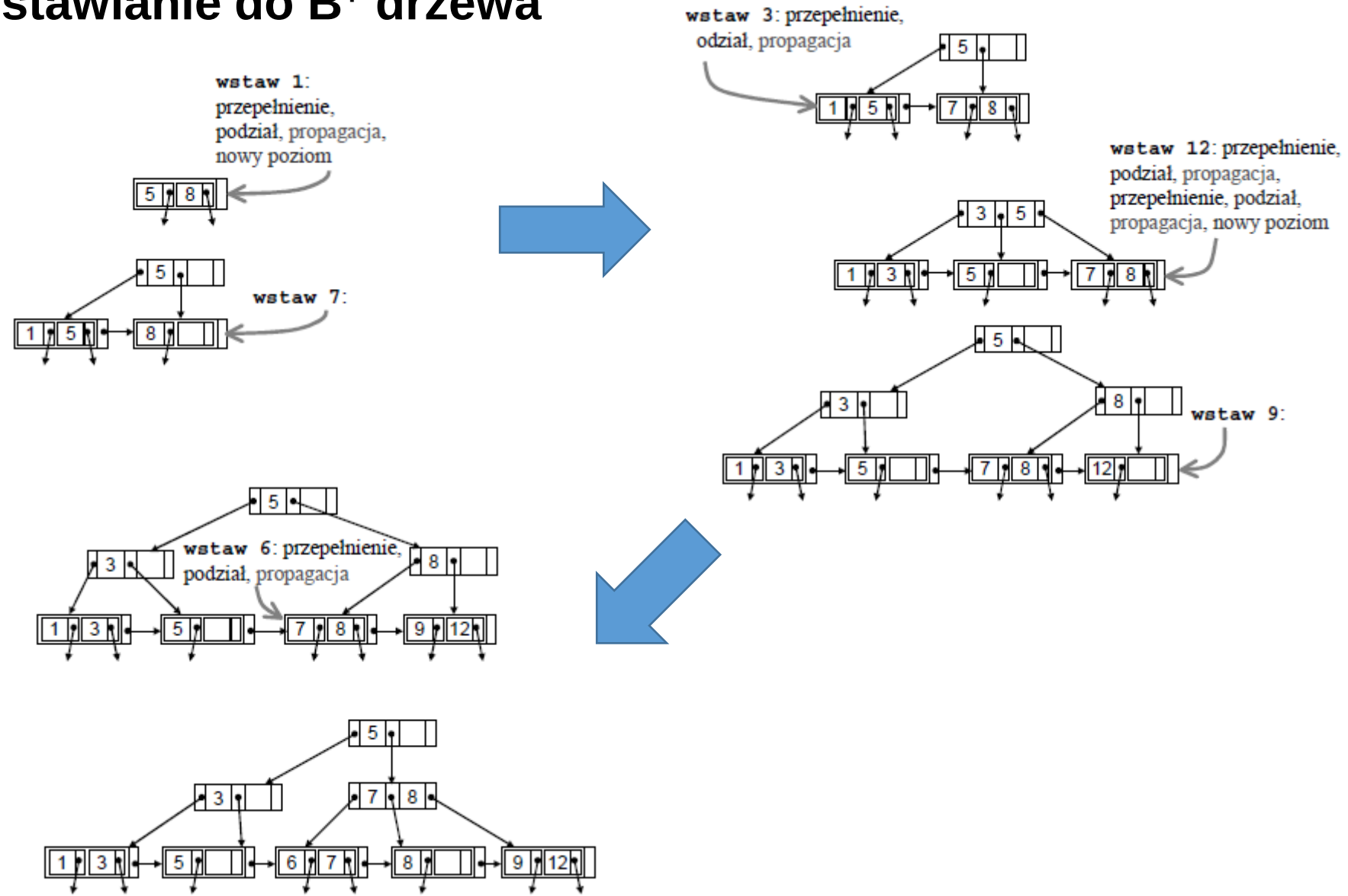
Przykładowe B+ drzewo



Zmienia się trochę procedura usuwania(bardziej) i dodawania(mniej)

Zadanie – utworzyć to drzewo (założyć, że m=1)

Wstawianie do B+ drzewa



www.cs.put.poznan.pl/mwojciechowski/slides/bdwykl/indeksy_cz3.pdf

B^+ drzewo - usuwanie

Procedura usuwania polega na usunięciu elementu tylko z liścia. Element występujący na wyższych poziomach nie musi być usuwany (służy on do nawigacji w B-drzewie)