CoLoSL

Concurrent Local Subjective Logic

Azalea Raad Jules Villard Philippa Gardner

Imperial College London

10 April 2015

Global Shared Resources

$$P_1 \wedge P_2$$

$$\frac{\{P1\}\ C1\ \{Q1\} \qquad \{P2\}\ C2\ \{Q2\}}{\{P1 \wedge P2\}\ C1 \parallel C2\ \{Q1 \wedge Q2\}}$$

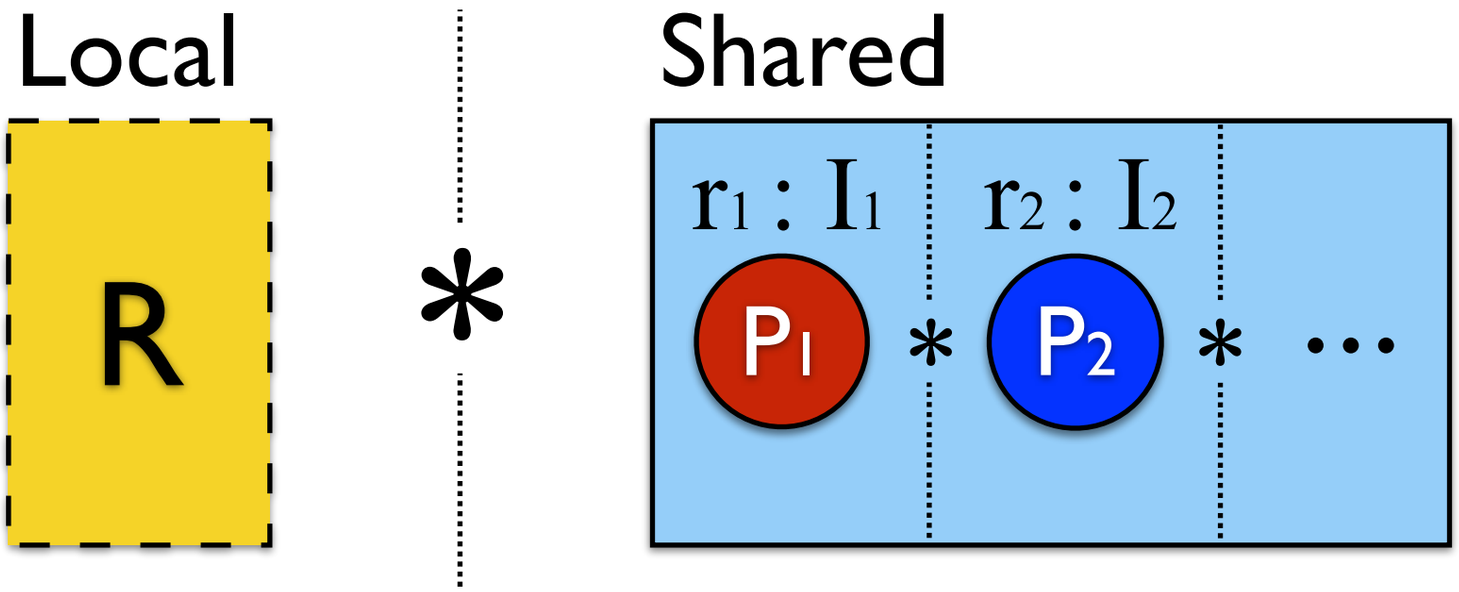
Owicki-Gries, Rely-Guarantee

Global Shared Resources

$$\frac{\{\boxed{P1}\}\ C1\ \{\boxed{Q1}\} \qquad \{\boxed{P2}\}\ C2\ \{\boxed{Q2}\}}{\{\boxed{P1 \wedge P2}\}\ C1 \parallel C2\ \{\boxed{Q1 \wedge Q2}\}}$$

- Shared Resources
  - **No framing** : reasoning on GLOBAL resources
- Interference
  - **No framing** : interference on ALL resources considered
- Extension
  - **No extension** : cannot dynamically share resources/extend interference

Disjoint Shared Resources



CSL, CAP, HOCAP, iCAP, TaDA

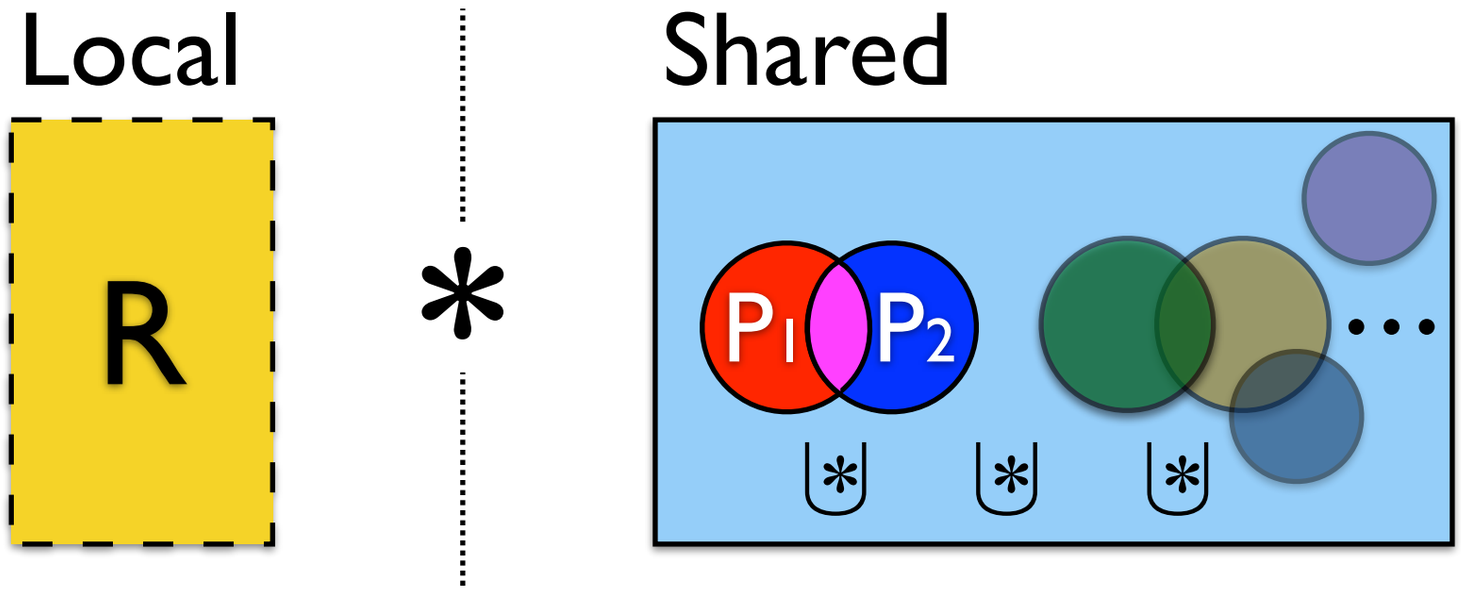
Disjoint Shared Resources

$$\frac{\{\boxed{P1}\}\ C1\ \{\boxed{Q1}\} \qquad \{\boxed{P2}\}\ C2\ \{\boxed{Q2}\}}{\{\boxed{P1} * \boxed{P2}\}\ C1 \parallel C2\ \{\boxed{Q1} * \boxed{Q2}\}}$$

- Shared resources / Interference
  - **Limited framing**:
    - **Static** (pre-determined) frames (regions/ invariants)
    - **Physically Disjoint** frames

- Extension
  - **Limited extension:**
    - Can create new regions/ invariants
    - **Cannot** extend regions with more resources/invariants

CoLoSL: Concurrent Local Subjective Logic



CoLoSL

CoLoSL: **Co**ncurrent **Lo**cal **S**ubjective **L**ogic

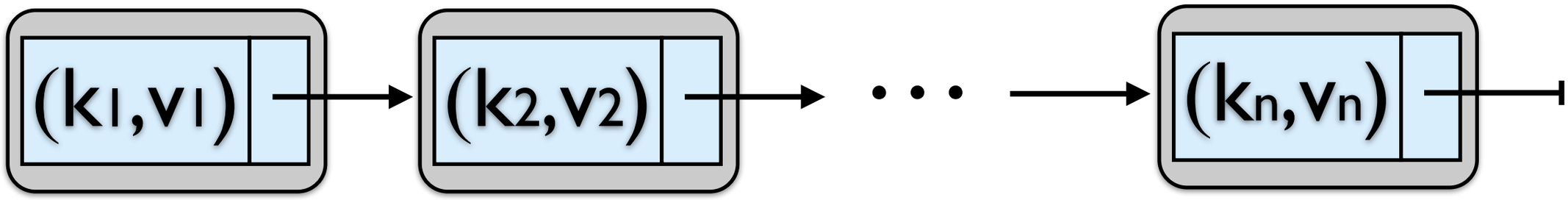
- Shared resources
  - **Dynamic** framing
  - **Overlapping** frames
- Interference
  - **Dynamic** framing/rewriting of interference
- Extension
  - **Dynamic** extension of shared state with new resource/interference

Why CoLoSL?

- **<u>Subjective/ Overlapping</u>** Shared Resources
  - Proof modularity; better abstraction

Ordered Singly Linked-List <K, V>

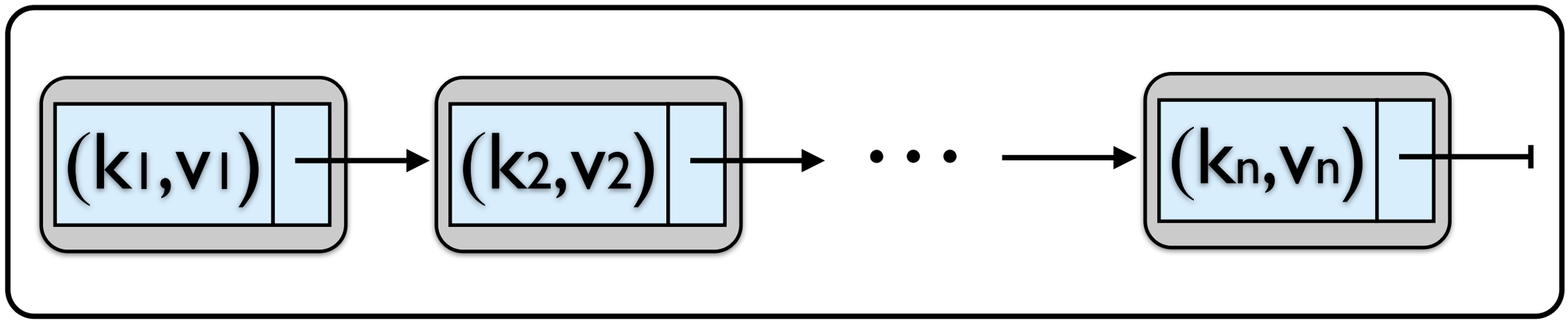
List([(k1, v1), (k2, v2), … , (kn, vn)]) =



- **insertion/removal** involves pointer surgery

Concurrent Ordered Singly Linked-List<K,V>

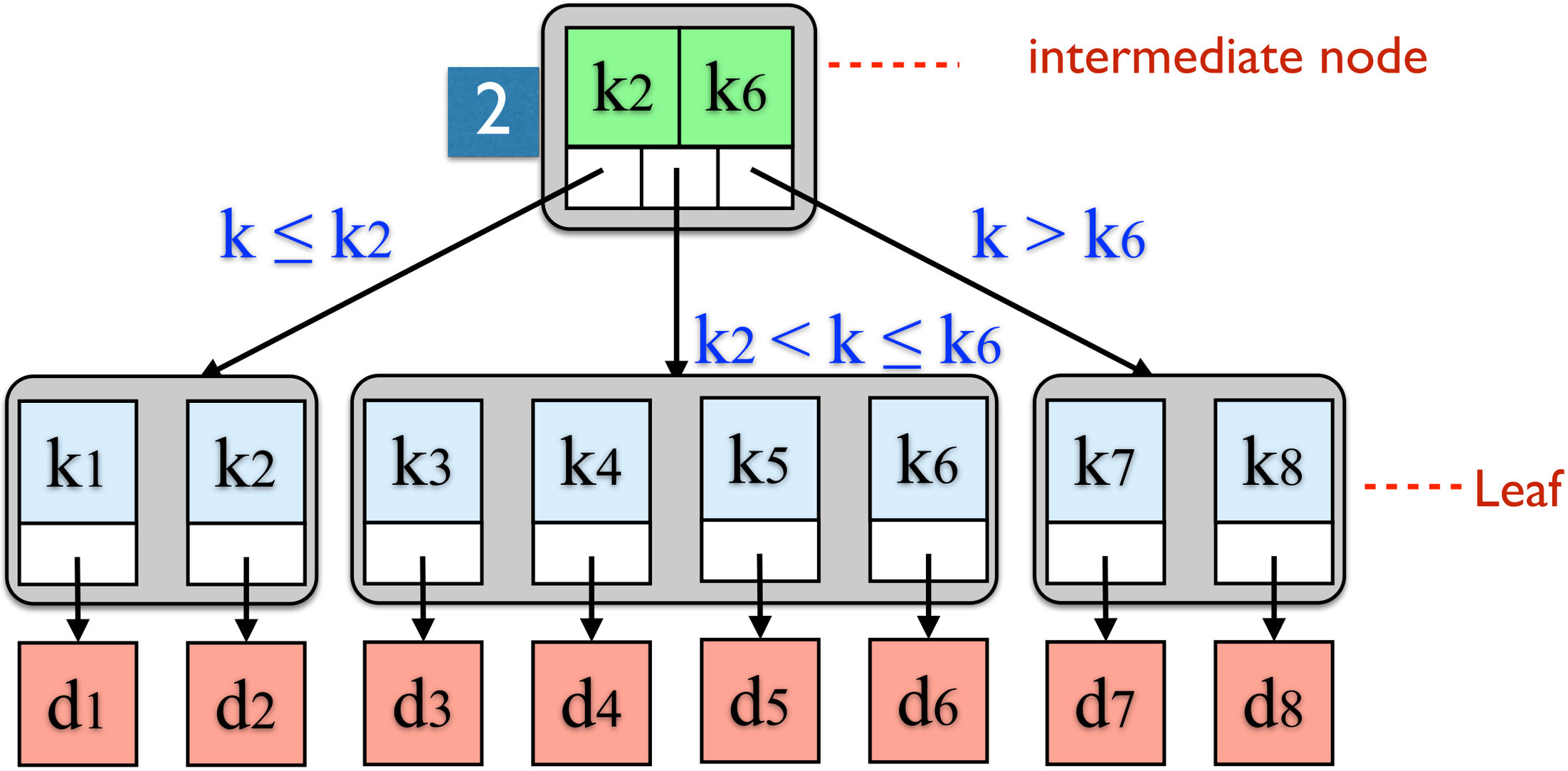
List([(k1, v1), (k2, v2), … , (kn, vn)) =



$I_L = I_{add} \cup I_{rem} \cup I_{map}$

- **<u>insertion/removal</u>** involves pointer surgery

Balanced Search Tree<K,V> (Degree 2)

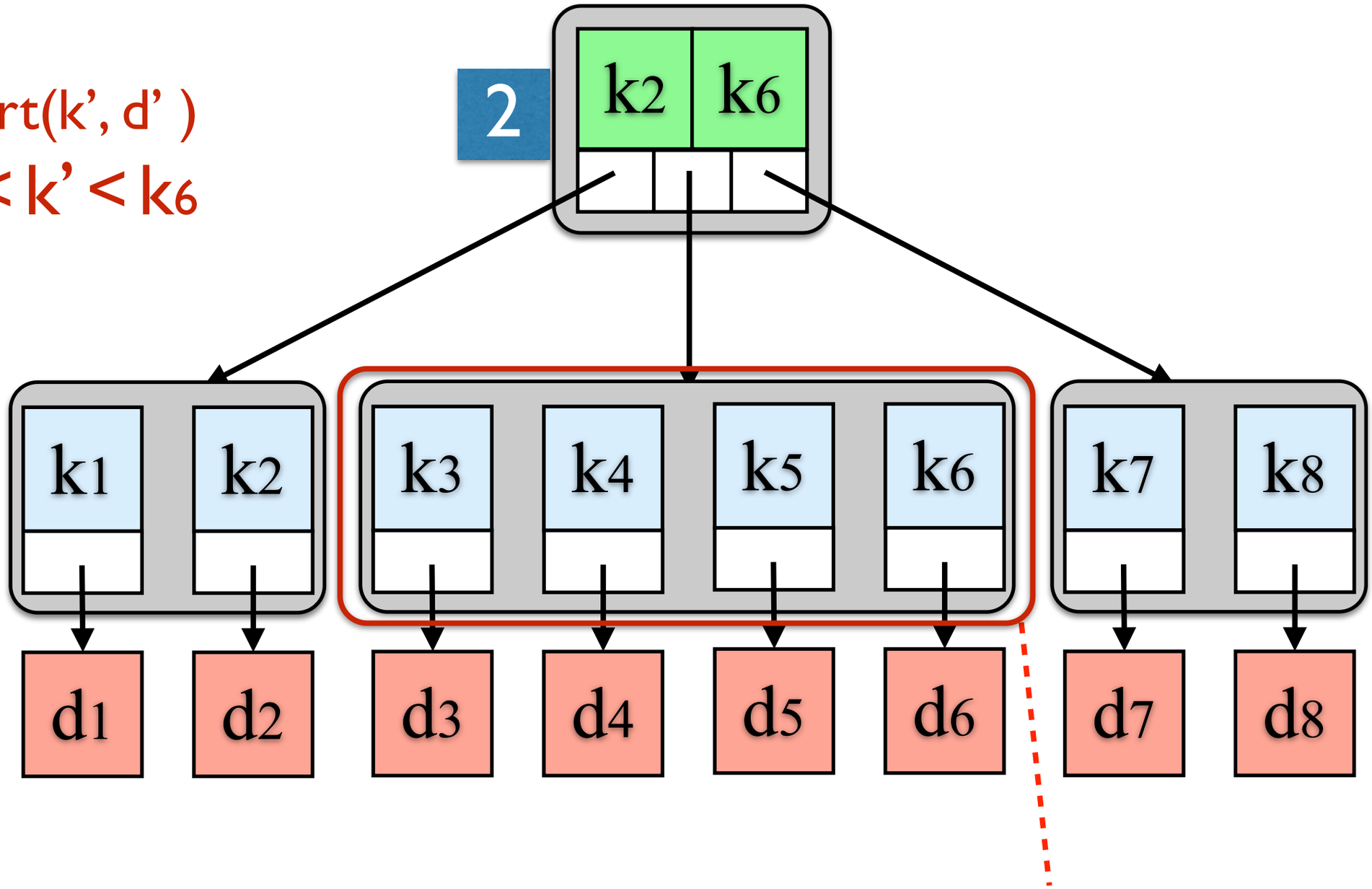


- **Balanced**: all immediate subtrees of a node have the same height
- **Leaf-heavy**: Data (values) stored in leaf nodes
- **Degree (d)**: no. of children (m) on each node $d \leq m \leq 2d$

Balanced Search Tree<K,V>

insert(k', d')

$k_5 < k' < k_6$

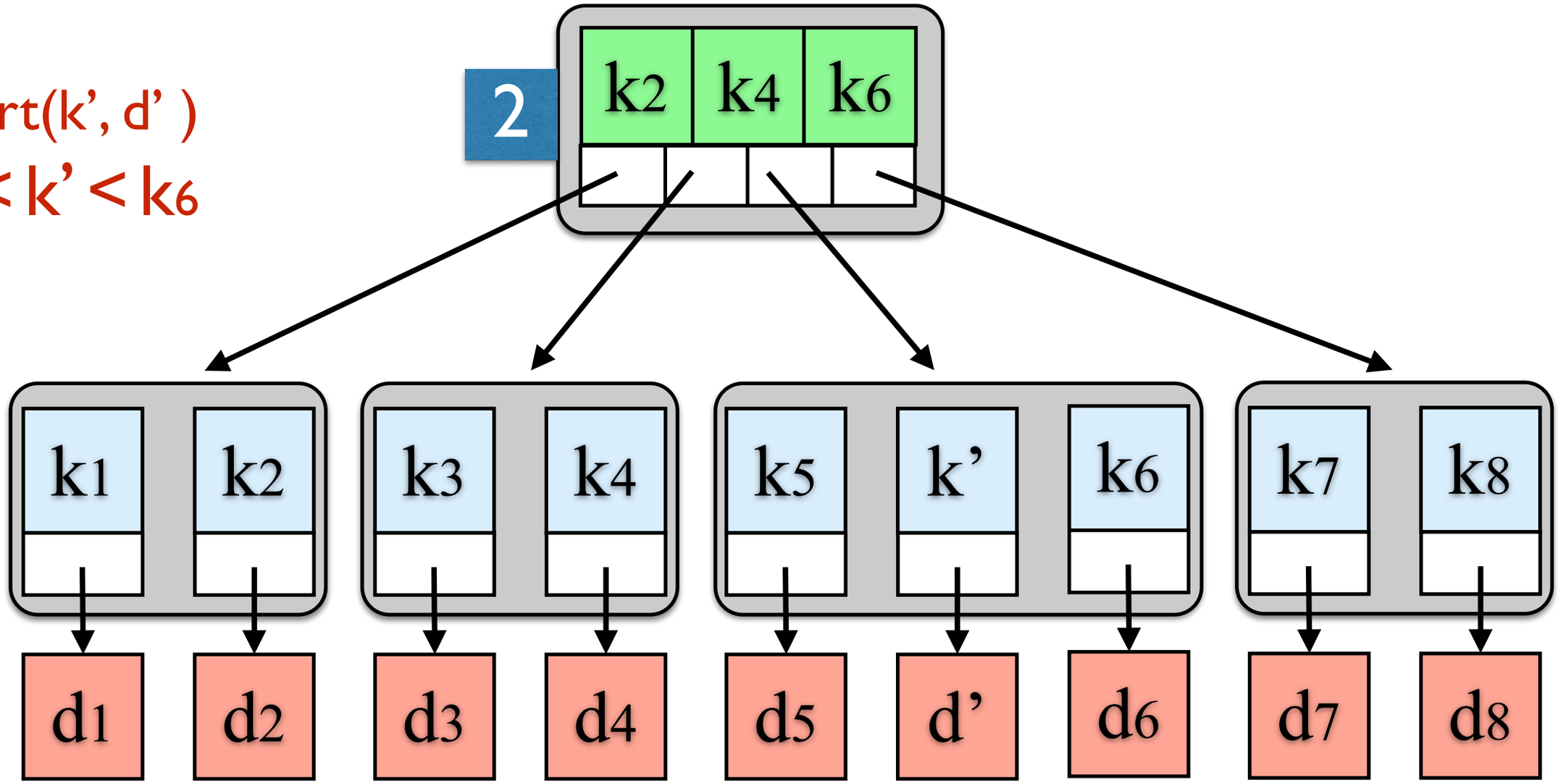


- **Insertion** may require **splitting** at max capacity

Balanced Search Tree<K,V>

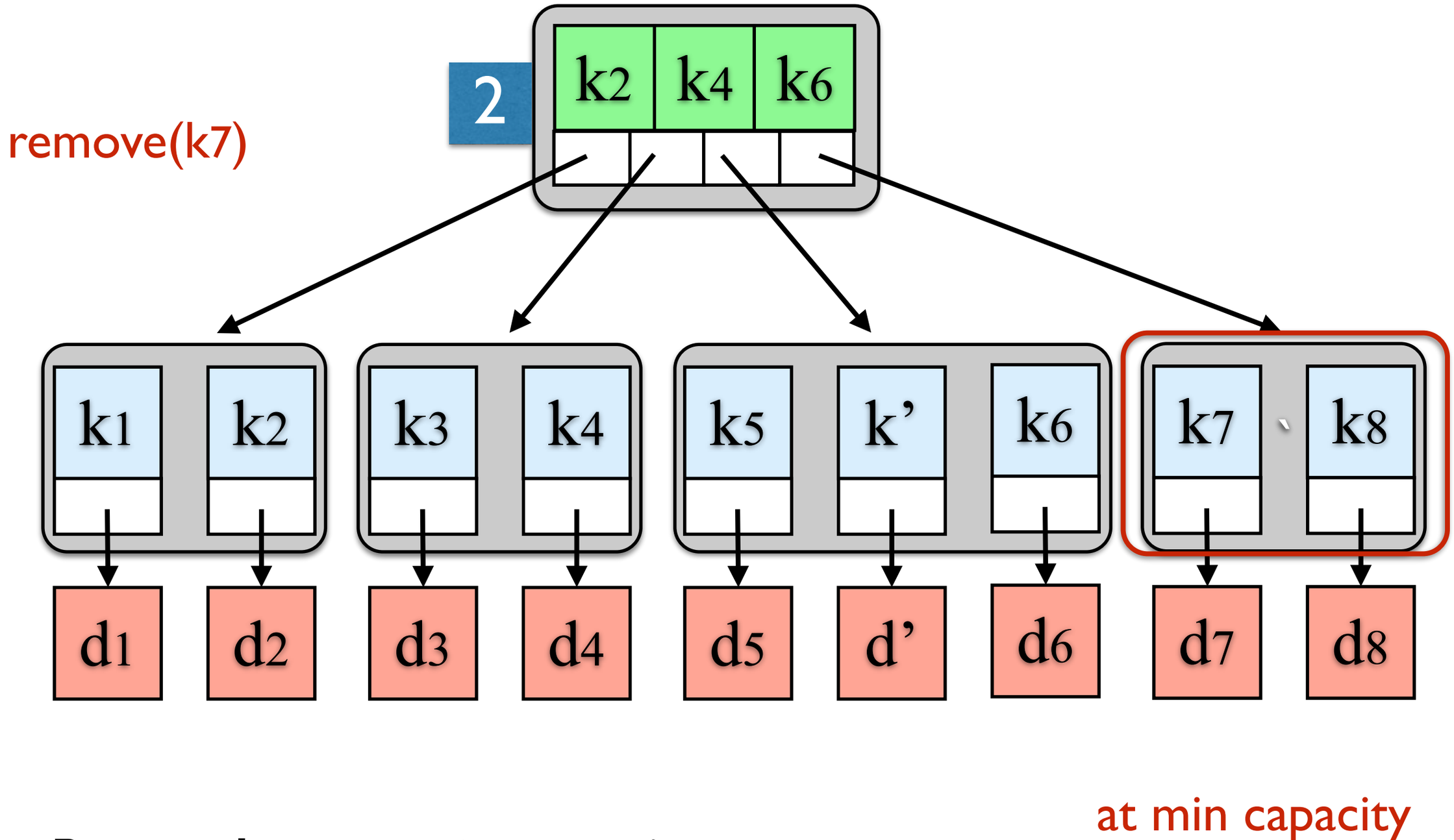
insert(k', d')

$k_5 < k' < k_6$



- **Insertion** may require **splitting**

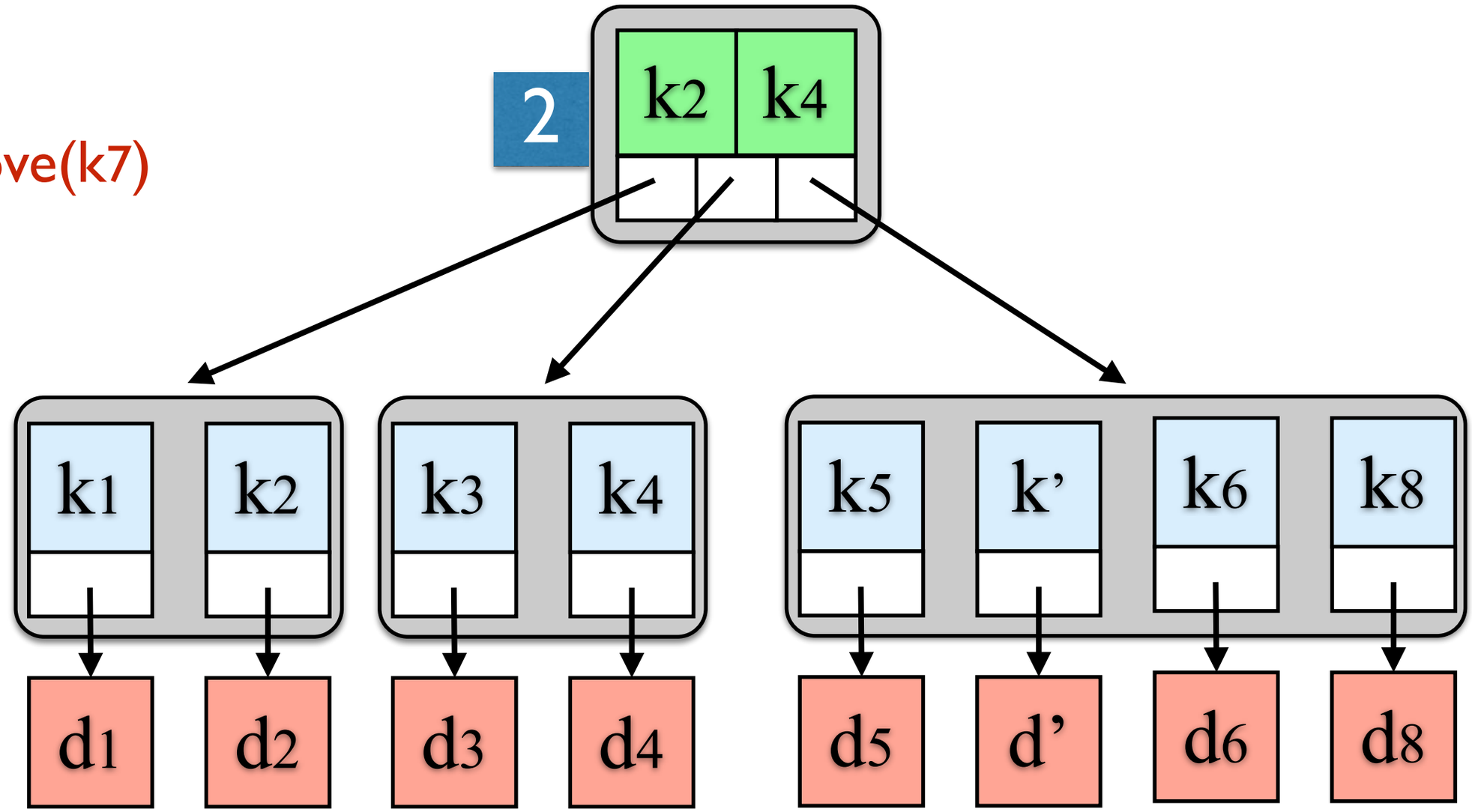
Balanced Search Tree<K,V>



- **Removal** may require **merging**

Balanced Search Tree<K,V>

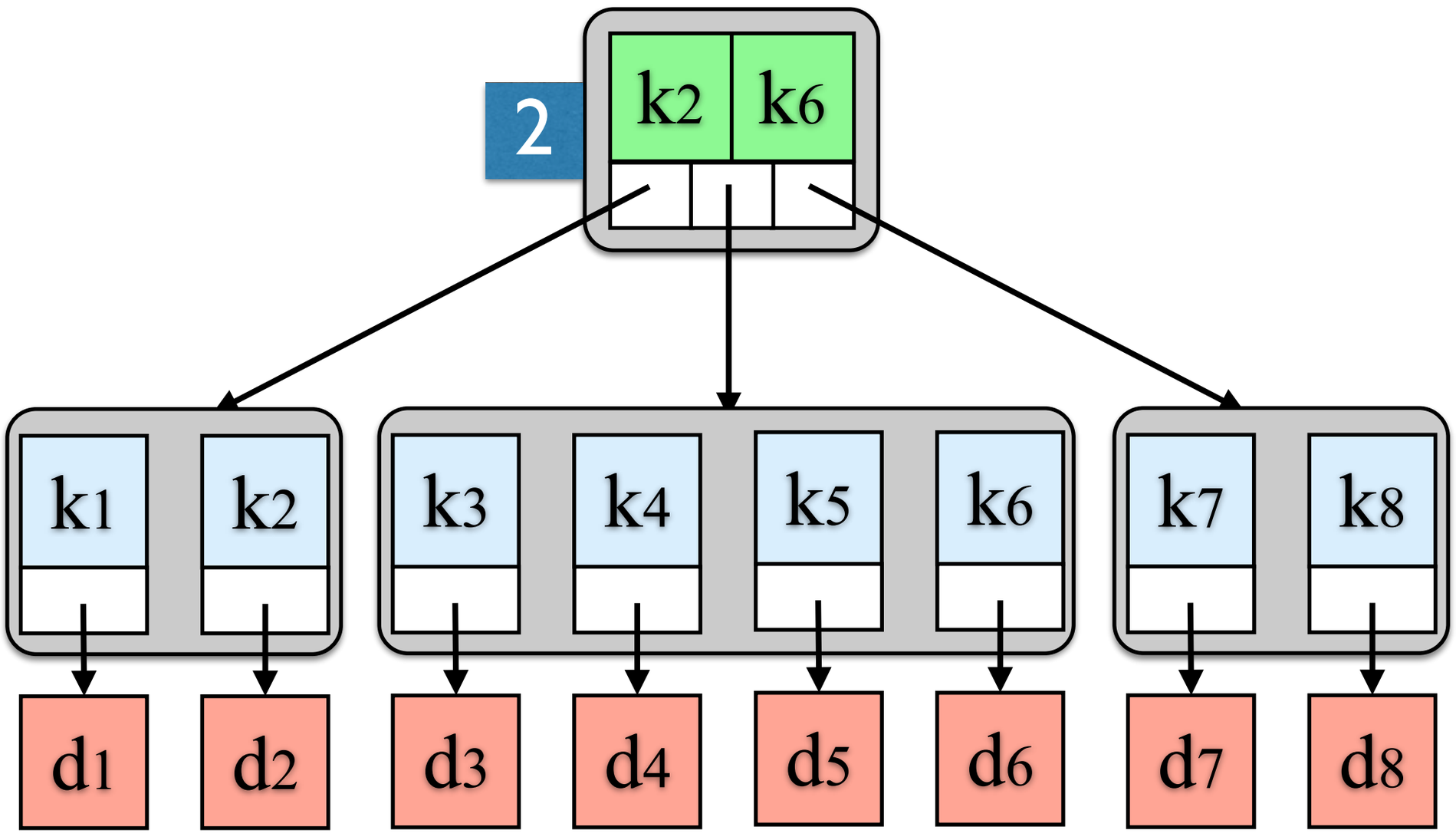
remove(k7)



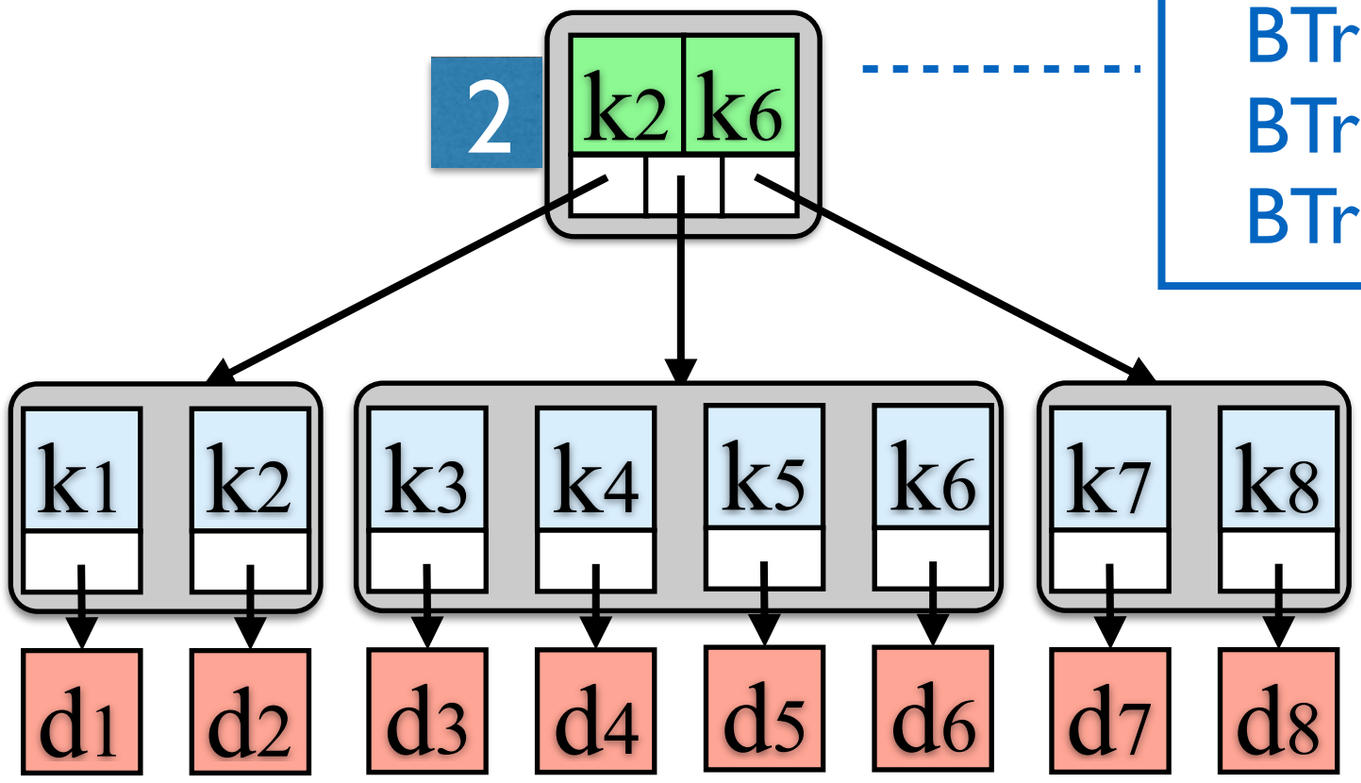
- **Removal** may require **merging**

Balanced Search Tree<K,V>

BTree 2 ([(k1, d1) … (k8, d8)]) =



Balanced Search Tree<K,V>



Node (k2, k6)
  BTree 1 [(k1,d1),(k2,d2)]
  BTree 1 [(k3,d3),(k4,d4),(k5,d5),(k6,d6)]
  BTree 1 [(k7,d7),(k8,d8)]

Leaf [(k7,d7),(k8,d8)]

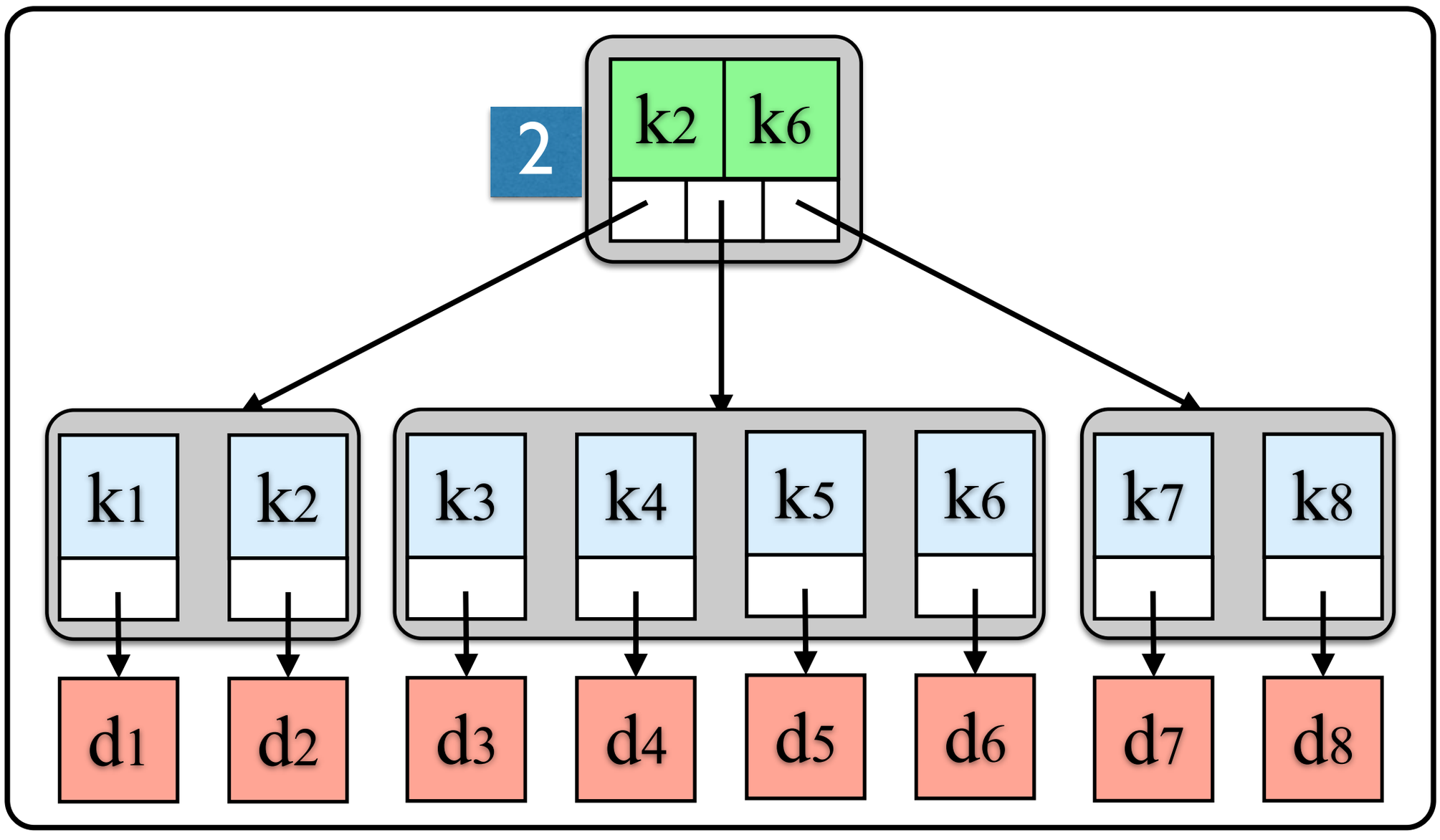
BTree (h+1) L =

  ∃ k1,…, km, L1, …, L(m+1) . L= L1 ⊎ … ⊎ L(m+1)
  Node (k1,…, km) (BTree h L1) … (BTree h L(m+1))

BTree 1 L = Leaf L

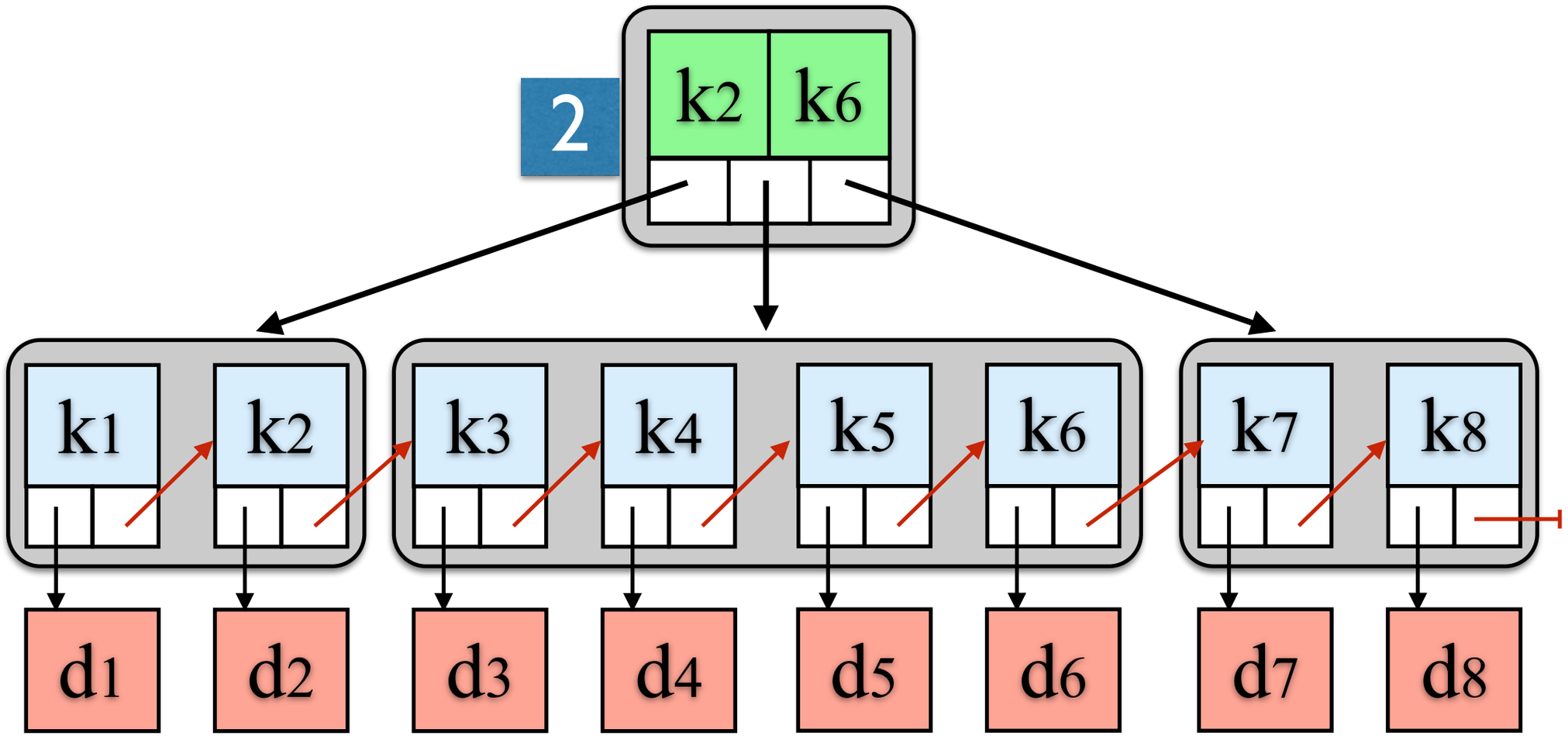
Concurrent Balanced Search Tree<K,V>

BTree([(k1, d1) … (k8, d8)]) =

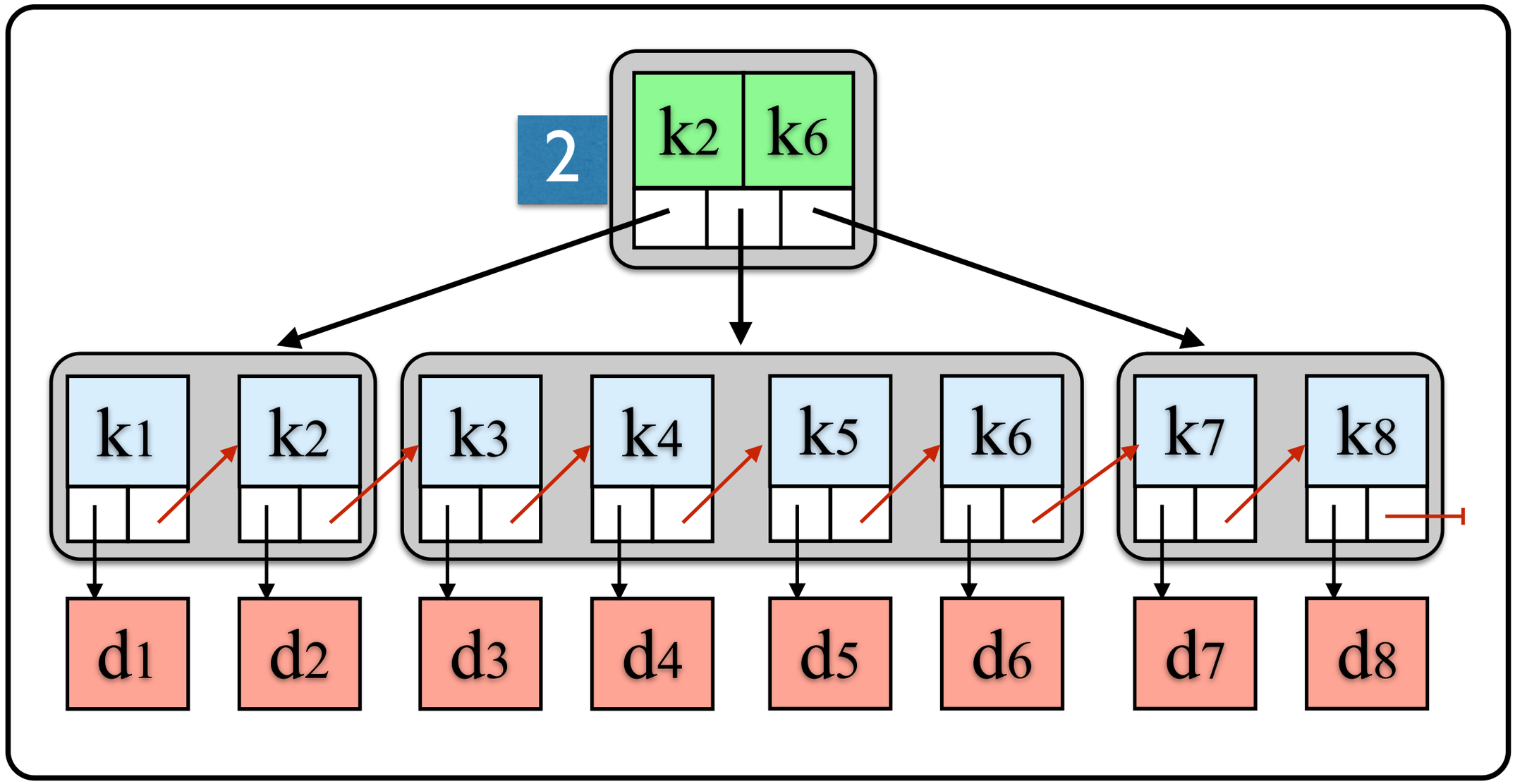


$I_B = I_{ins} \cup I_{del} \cup I_{srch}$

B+ Tree <K,V>



Concurrent B+ Tree <K,V>

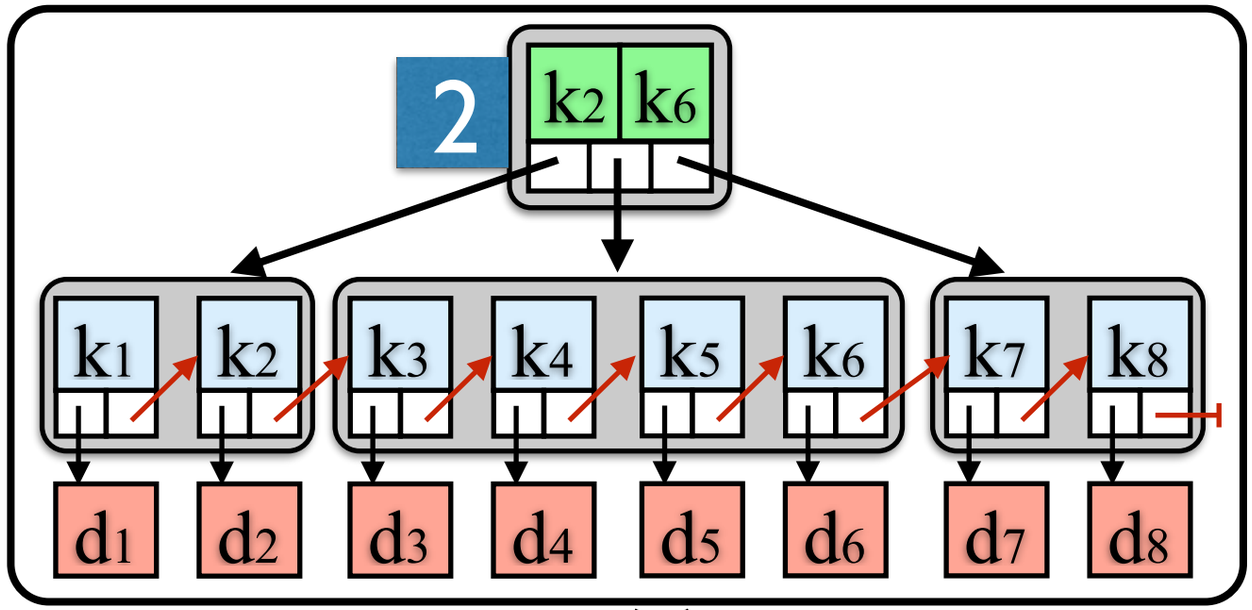


$I = I_B \cup I_L$

Concurrent B+ Tree <K,V> Wish List

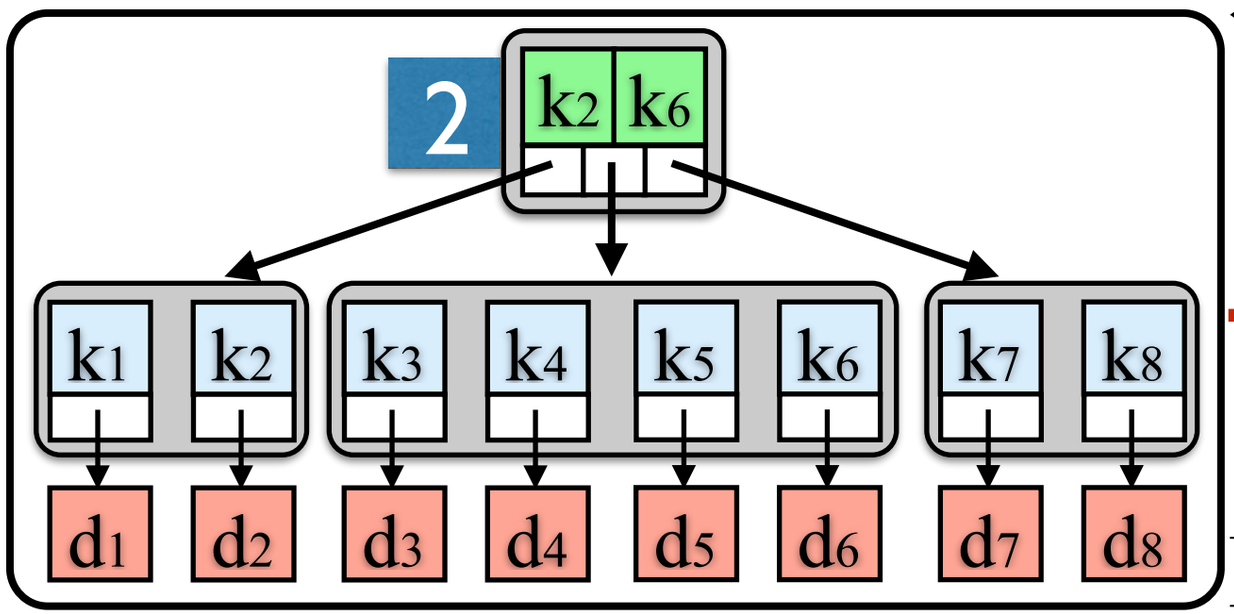
- **Module Composition**

B+ Tree h L $_{I_B \cup I_L}$ ⇔ BTree h L $_{I_B}$? + List L $_{I_L}$



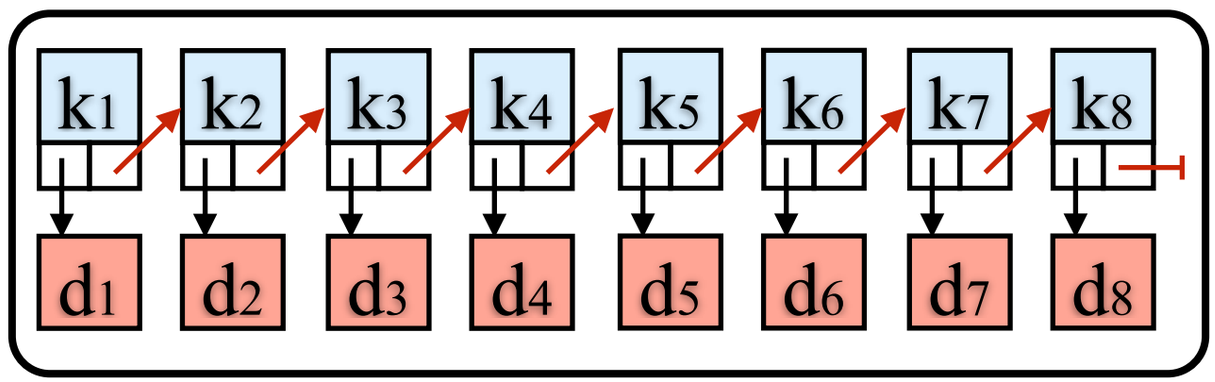
$I = I_B \cup I_L$

⇔



I_B

?+

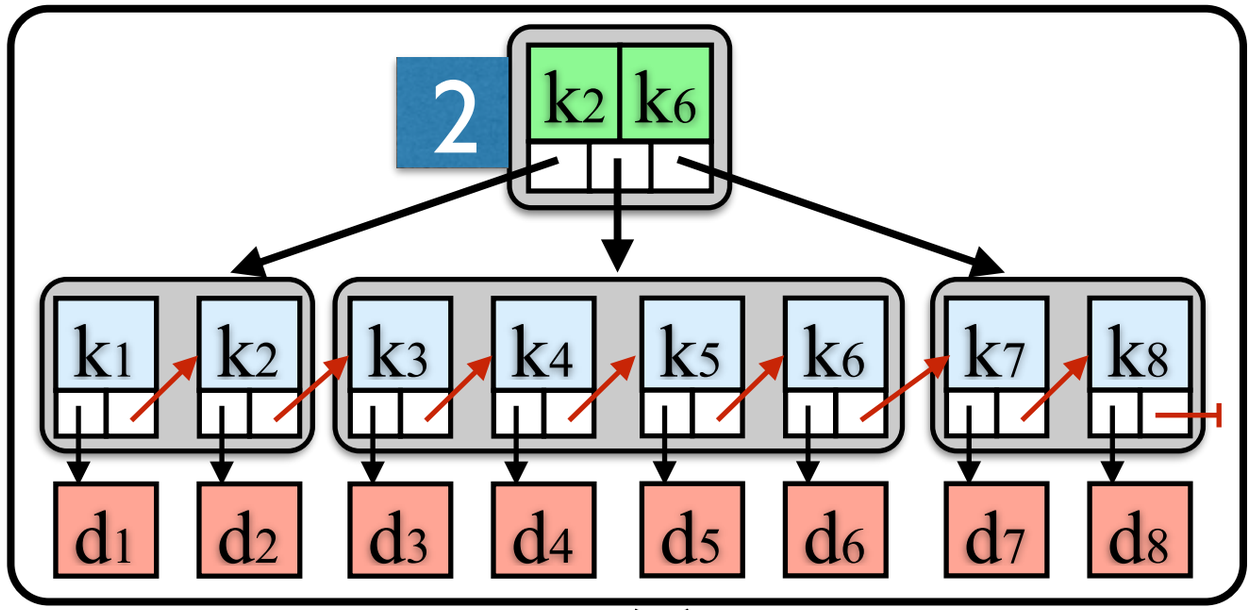


I_L

Concurrent B+ Tree <K,V>
(Existing Approaches)

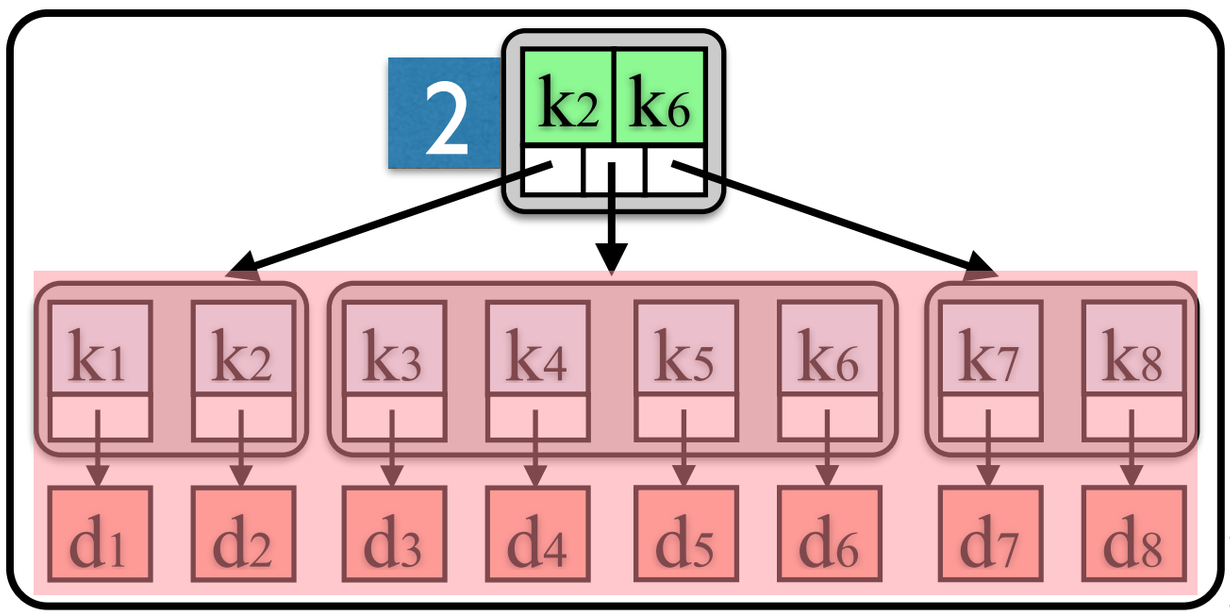
- **Module Composition**

B+ Tree h L $_{I_B \cup I_L}$ ⇔ BTree h L $_{I_B}$ * List L $_{I_L}$



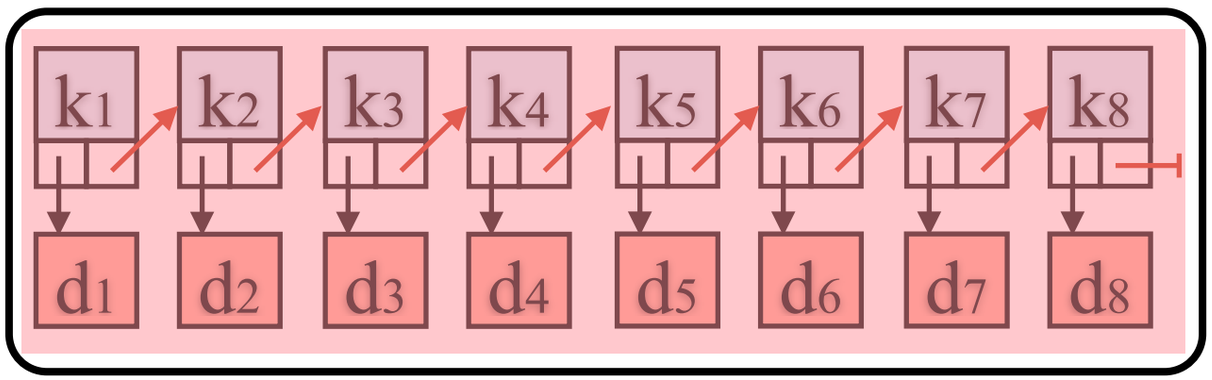
$I = I_B \cup I_L$

⇔ ✗



I_B

*

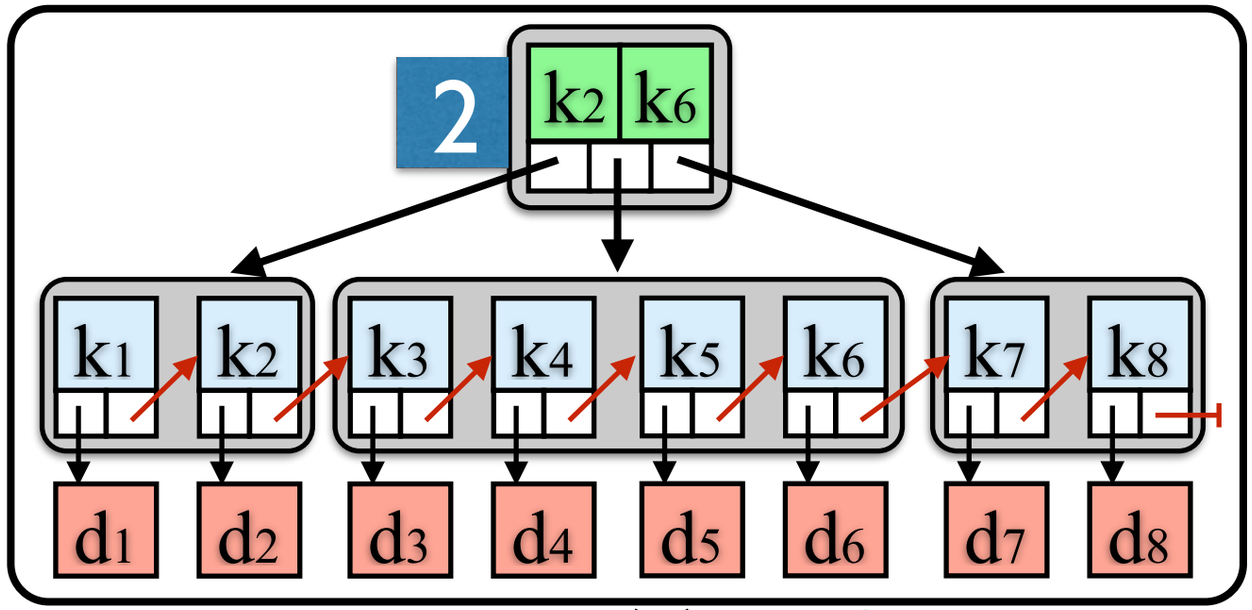


I_L

Concurrent B+ Tree <K,V>
(CoLoSL)

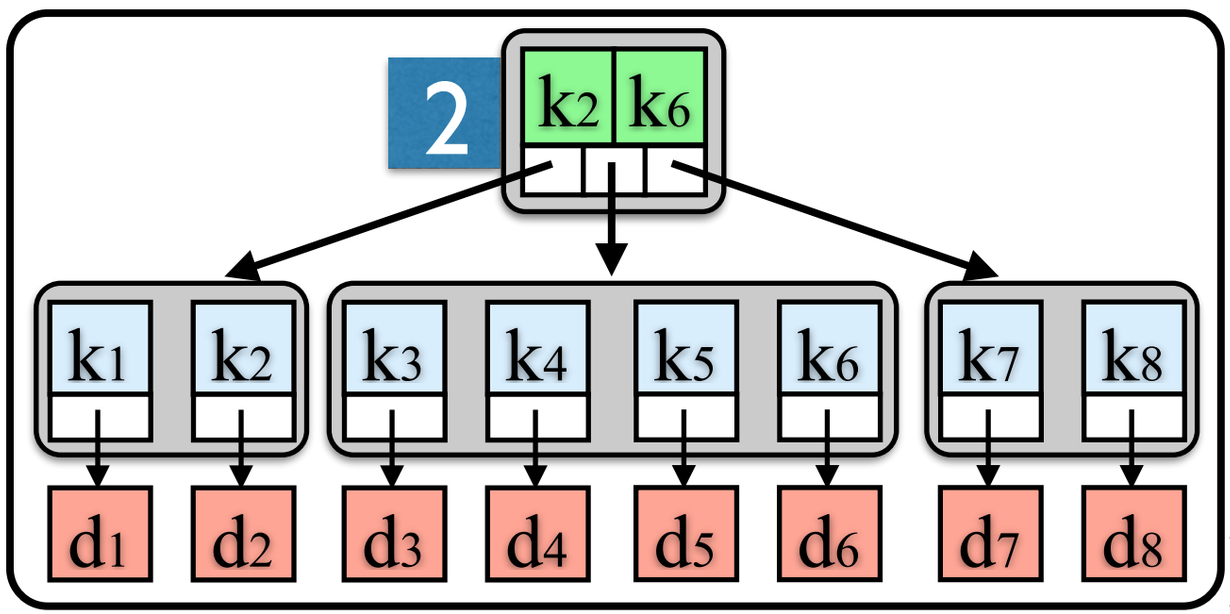
- **Module Composition**

B+ Tree h L $_{I_B \cup I_L}$ ⇔ BTree h L $_{I_B}$ ✻ List L $_{I_L}$



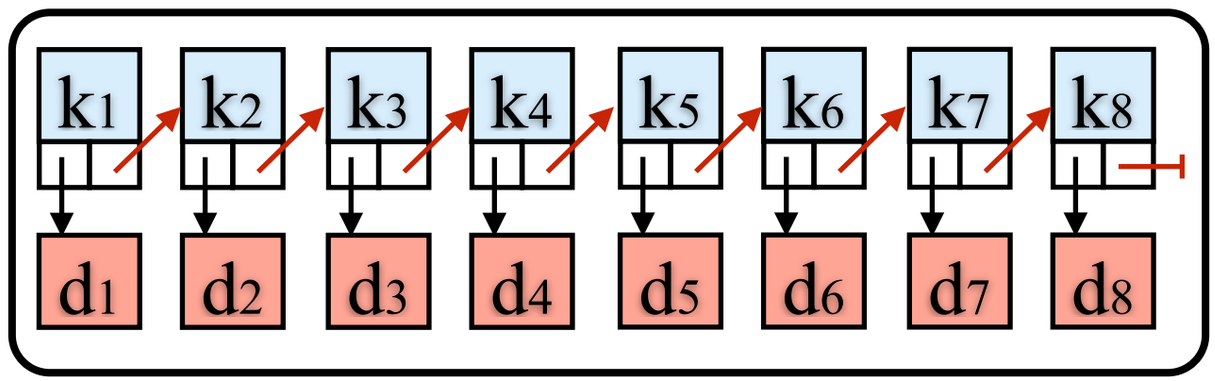
$I = I_B \cup I_L$

⇔ ✔



I_B

✻



I_L

B+ Tree <K,V> Wish List

- **Module Composition**

  $\boxed{\text{B+ Tree h L}}_{I_B \cup I_L} \Leftrightarrow \boxed{\text{BTree h L}}_{I_B} \text{ ✻ } \boxed{\text{List L}}_{I_L}$

- **Proof Modularity (overlapping frames)**
  - **Reuse** list-only operation (e.g. map)

    $\boxed{\text{BTree h L}}_{I_B} \text{ ✻ } \boxed{\text{List L}}_{I_L} \xRightarrow{\text{(frame)}} \boxed{\text{List L}}_{I_L}$

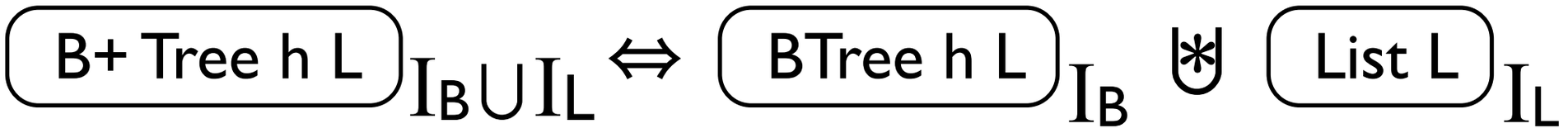
    $\{ \boxed{\text{List L}}_{I_L} \}$ `inc_all()` $\{ \boxed{\text{List inc(L)}}_{I_L} \}$

  - **Reuse** tree-only operation (e.g. search)

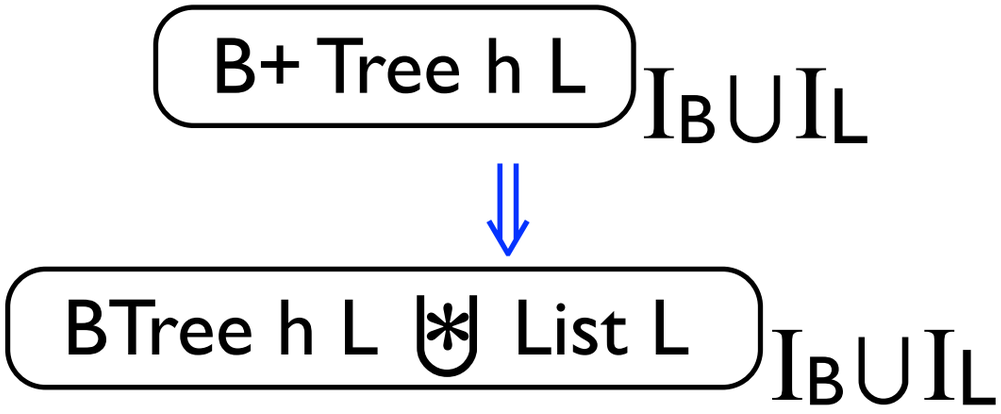
    $\boxed{\text{BTree h L}}_{I_B} \text{ ✻ } \boxed{\text{List L}}_{I_L} \xRightarrow{\text{(frame)}} \boxed{\text{BTree h L}}_{I_B}$

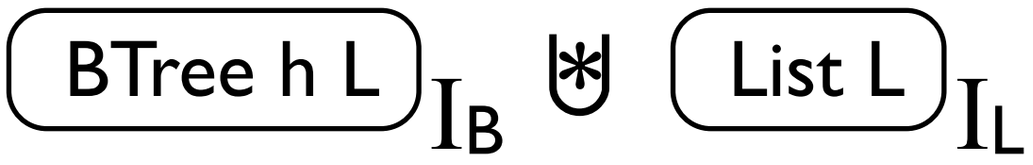
  - **Combine** tree/list module operations to implement B+ tree operations (e.g. remove)

CoLoSL Principles



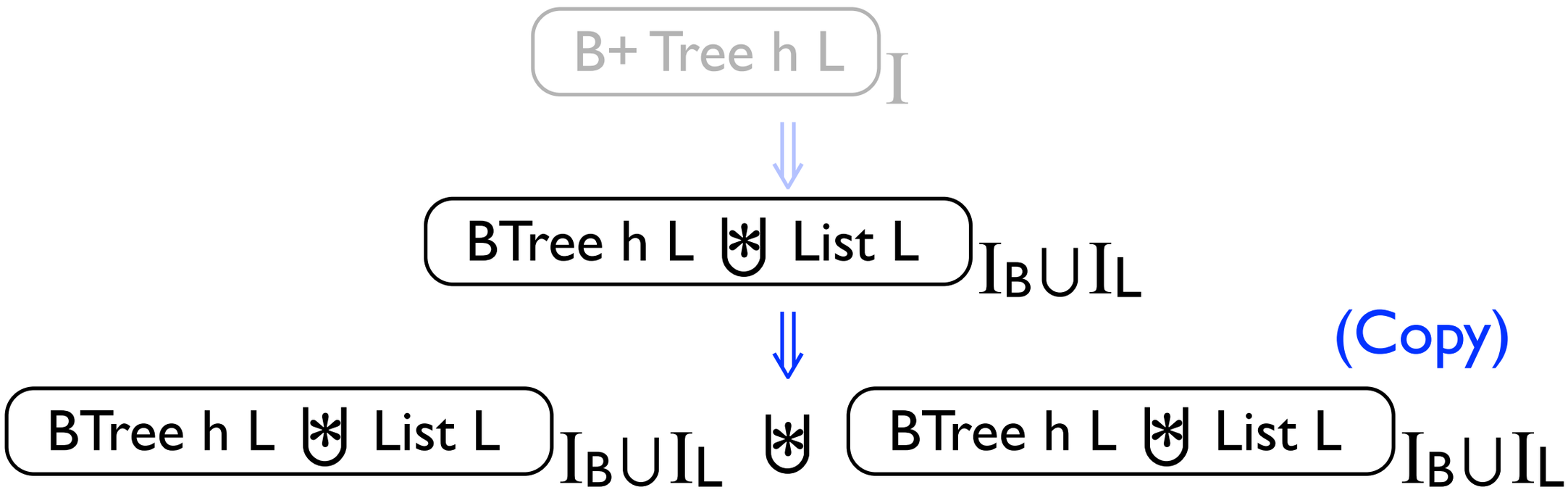
CoLoSL Principles



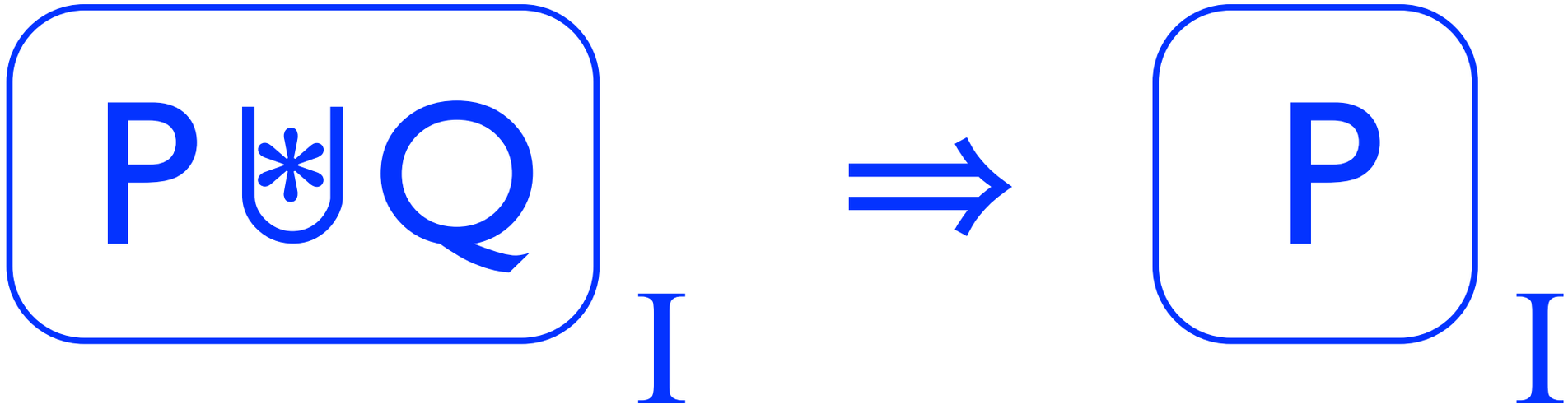


Duplicating Resources

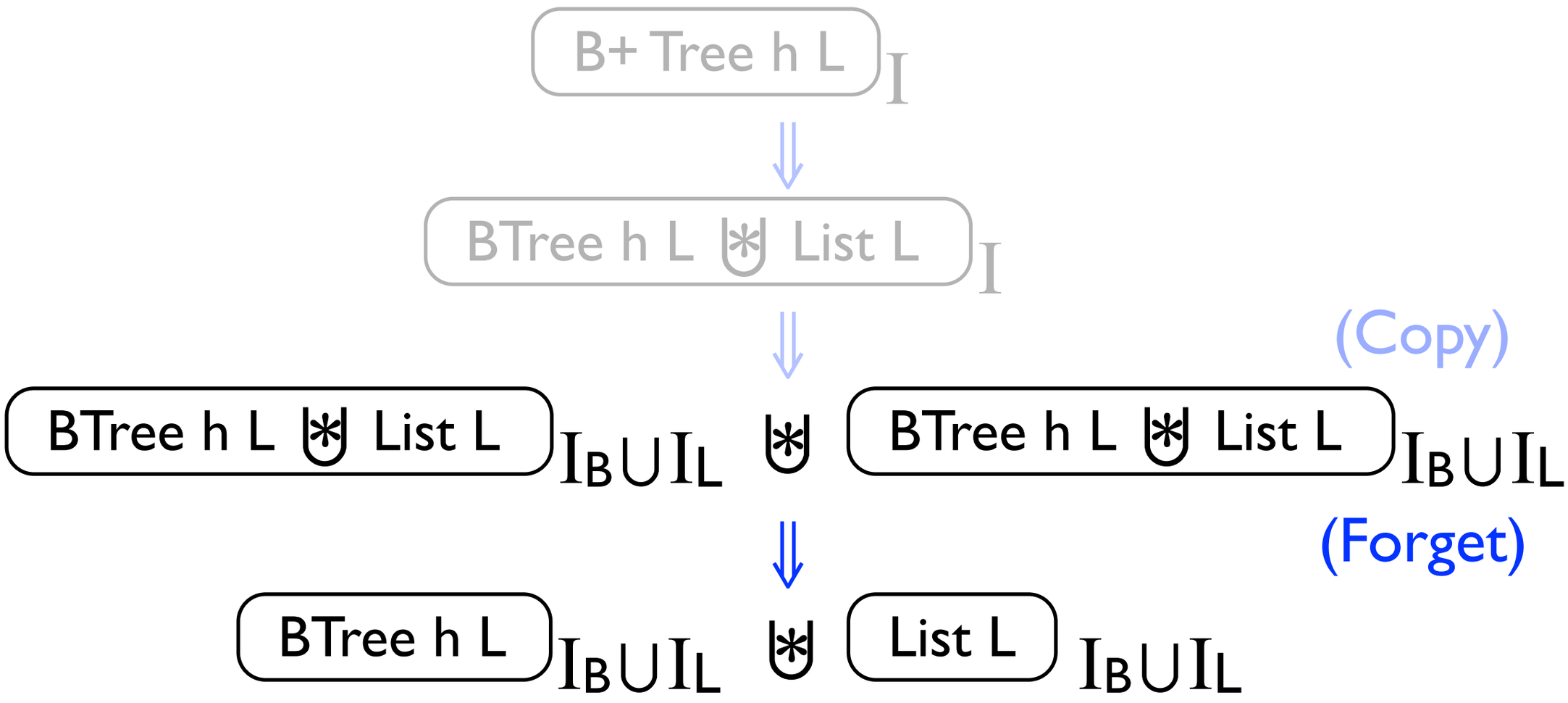
CoLoSL Principles



Forgetting Resources



CoLoSL Principles

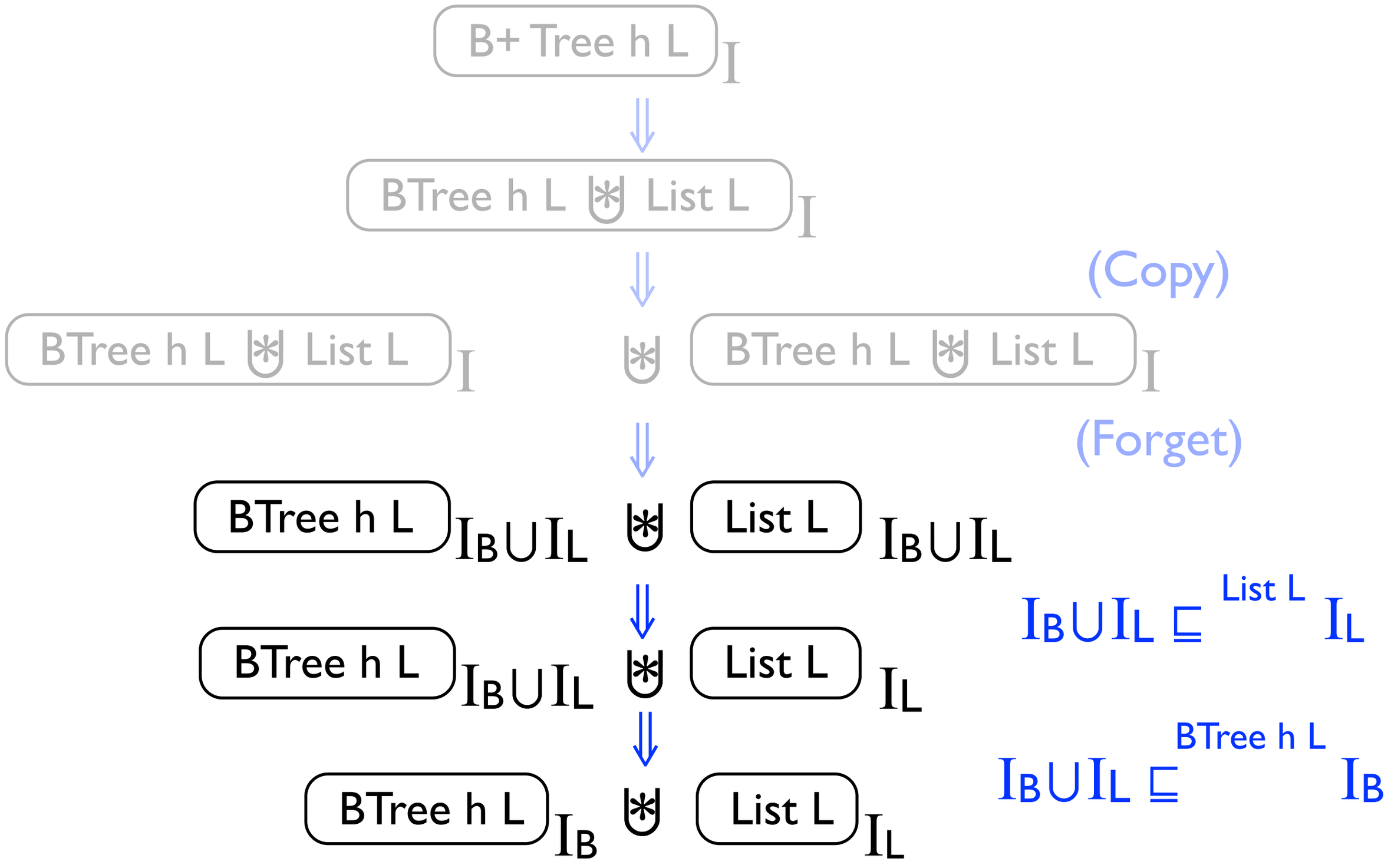


Forgetting Interference (Shift)

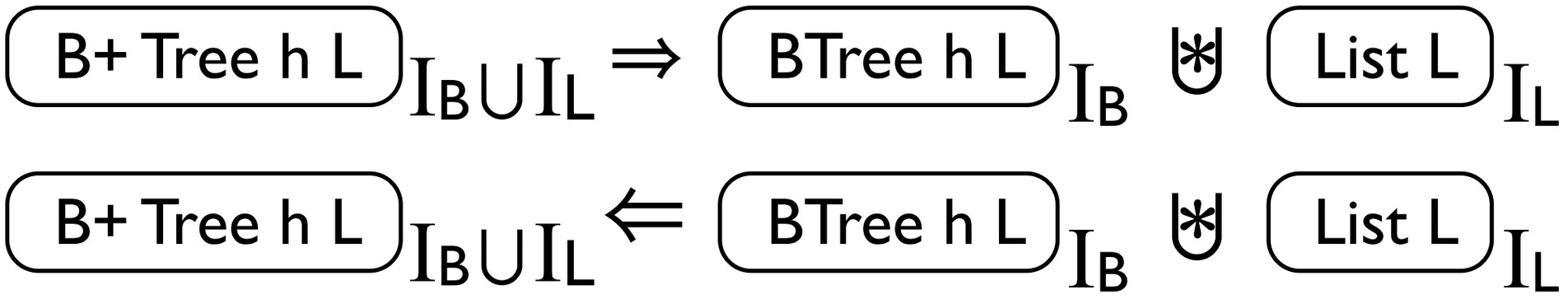
if $I \sqsubseteq^{P} I'$

then $\boxed{P}_{I} \Rightarrow \boxed{P}_{I'}$

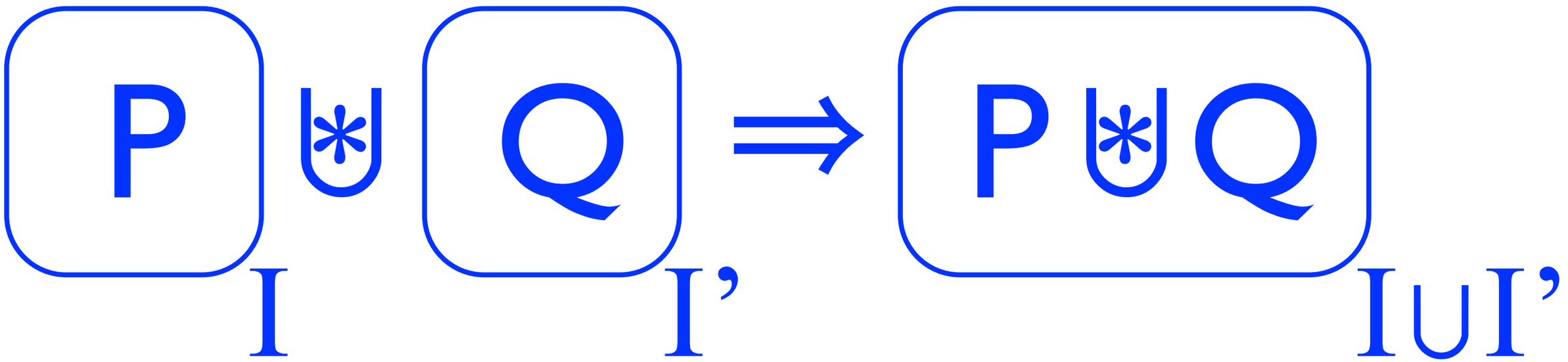
CoLoSL Principles



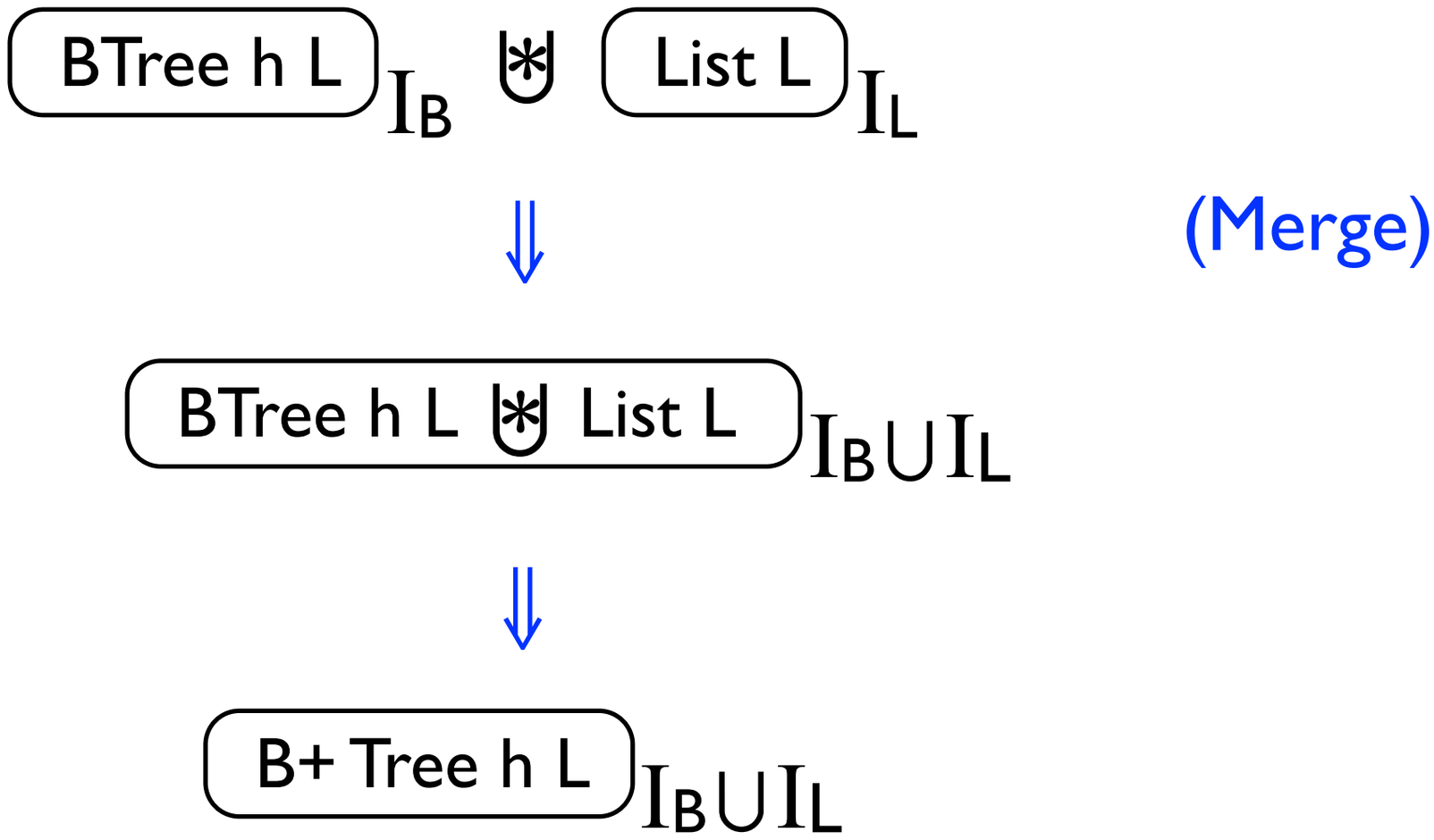
CoLoSL Principles



Merging Resources



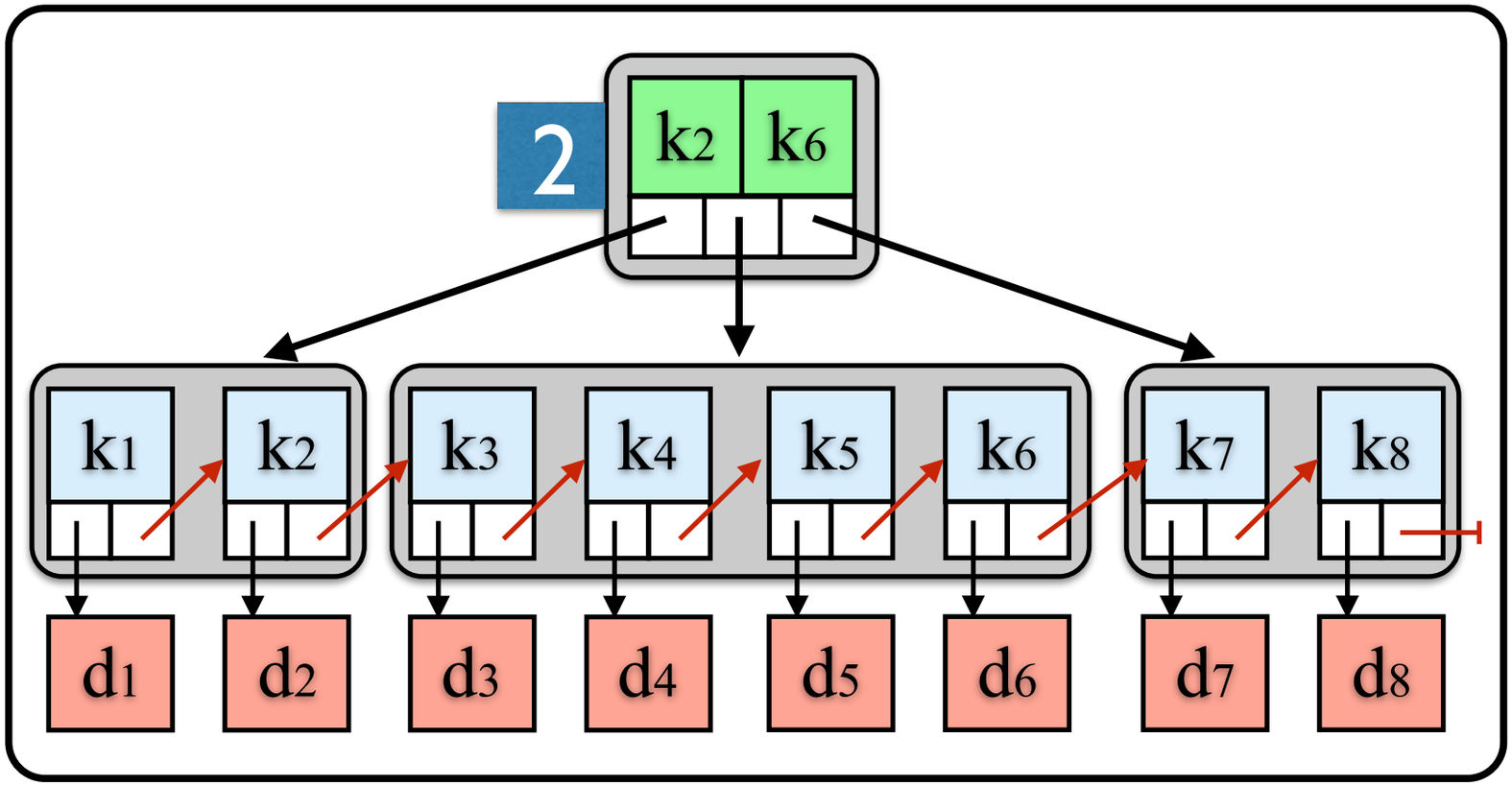
CoLoSL Principles



Why CoLoSL?

- **Subjective/ Overlapping** Shared Resources
  - Proof modularity; better abstraction
- **Local** Proofs
  - Proof reuse - proofs done for the largest possible context

B+ Tree <K,V>: Possible Extension



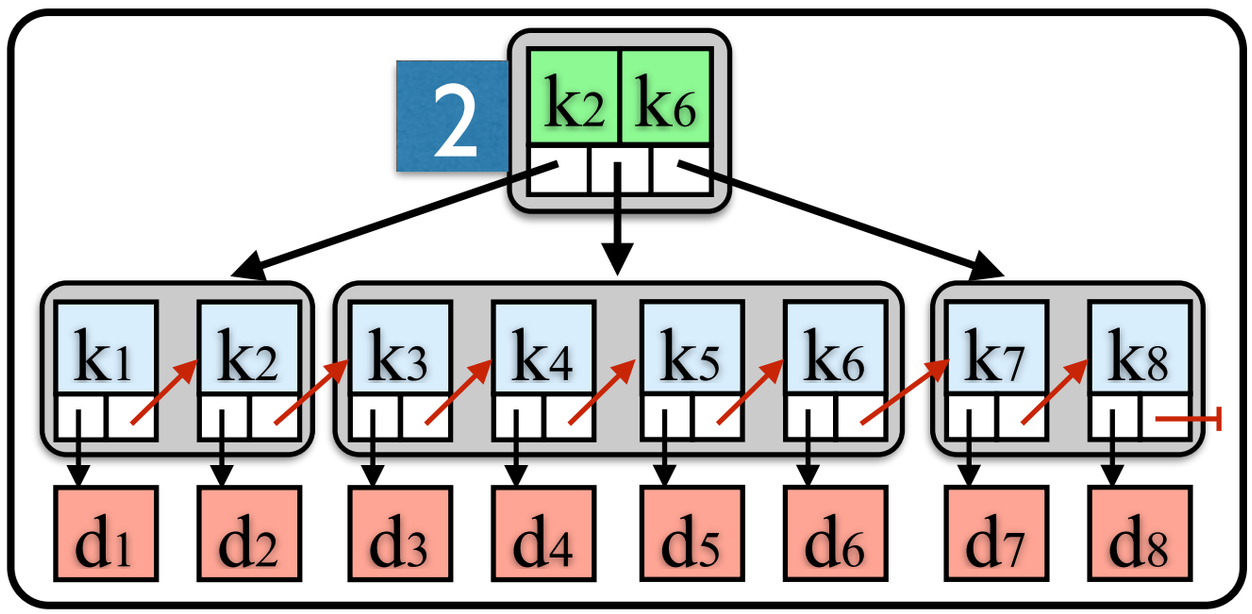
I ∪ Ideg

- Re-degree the tree
  - periodically change the degree (d) for better search time
- Re-degreeing ONLY affects the tree structure
- Re-degreeing does NOT affect the list structure
  - Should not have to reprove list operations

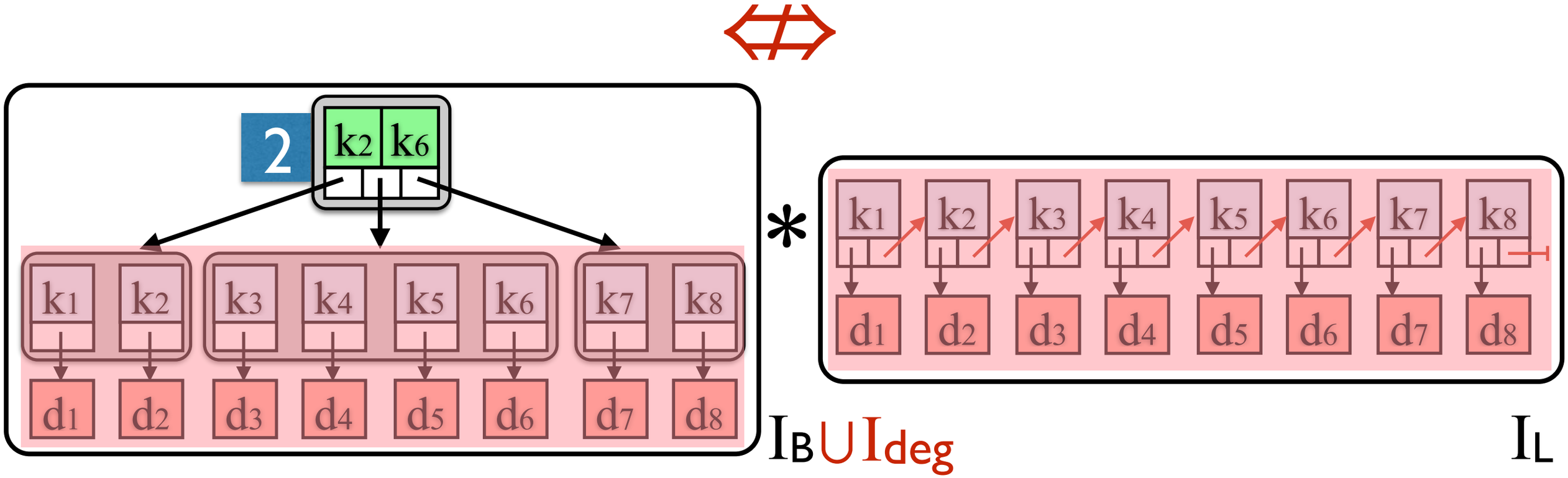
Ideg : 2 ⇝ 3

B+ Tree <K,V>: Possible Extension

(Existing Approaches)

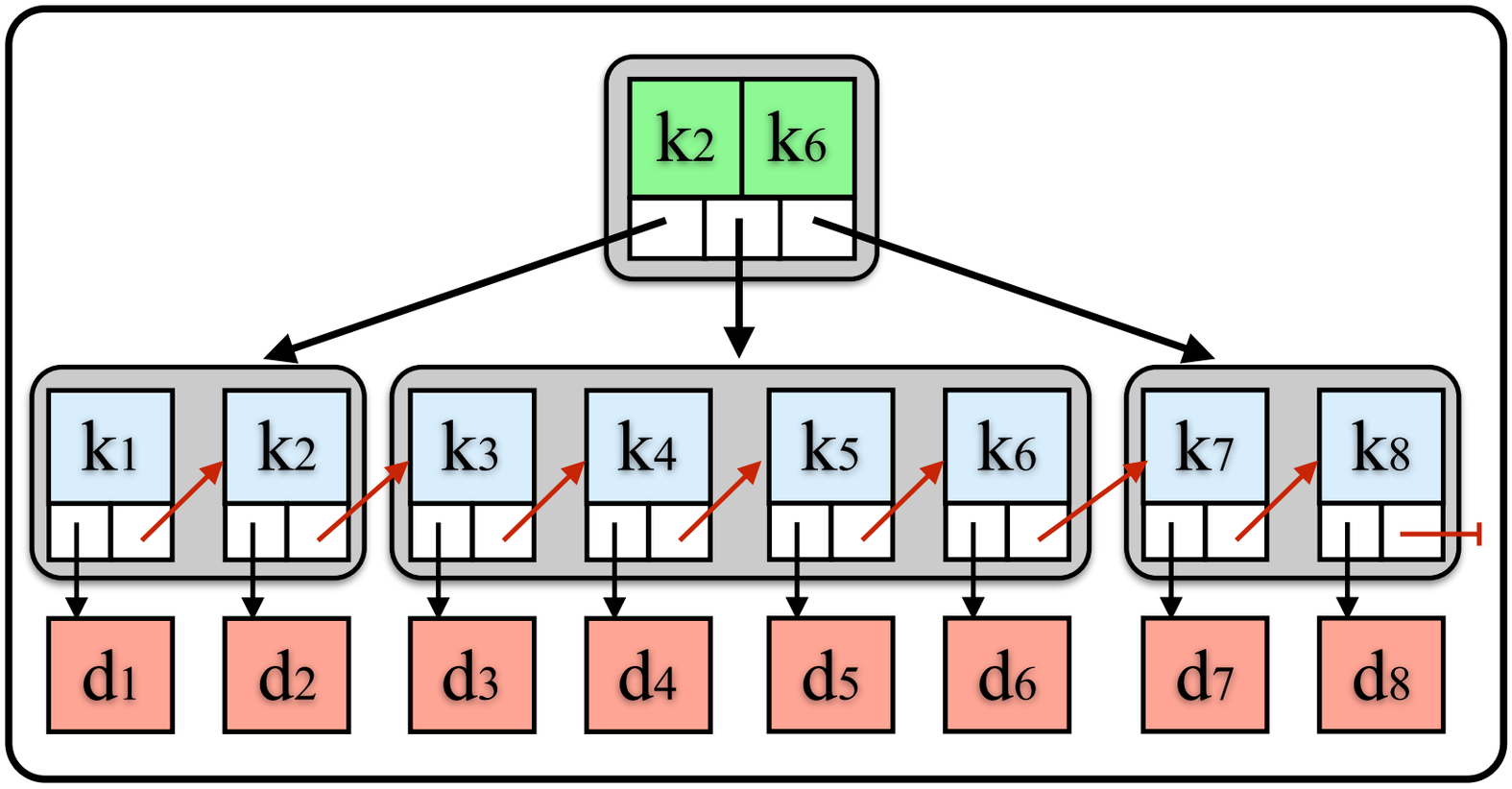


$I \cup I_{deg}$



$I_B \cup I_{deg}$ * I_L

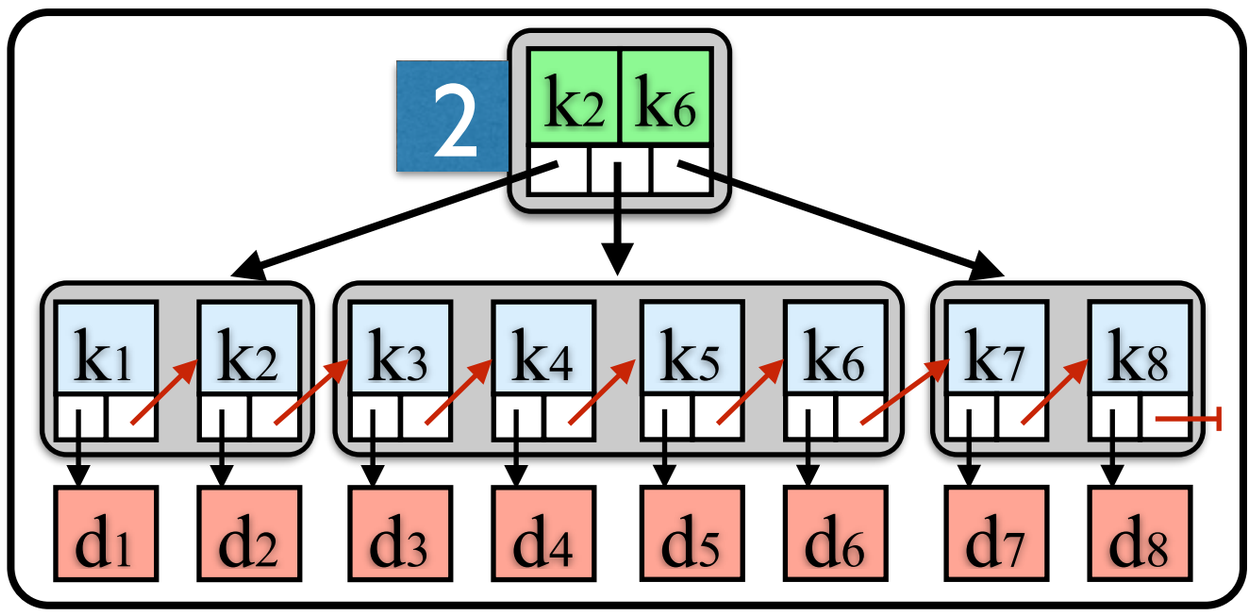
B+ Tree <K,V>: Possible Extension
(Existing Approaches)



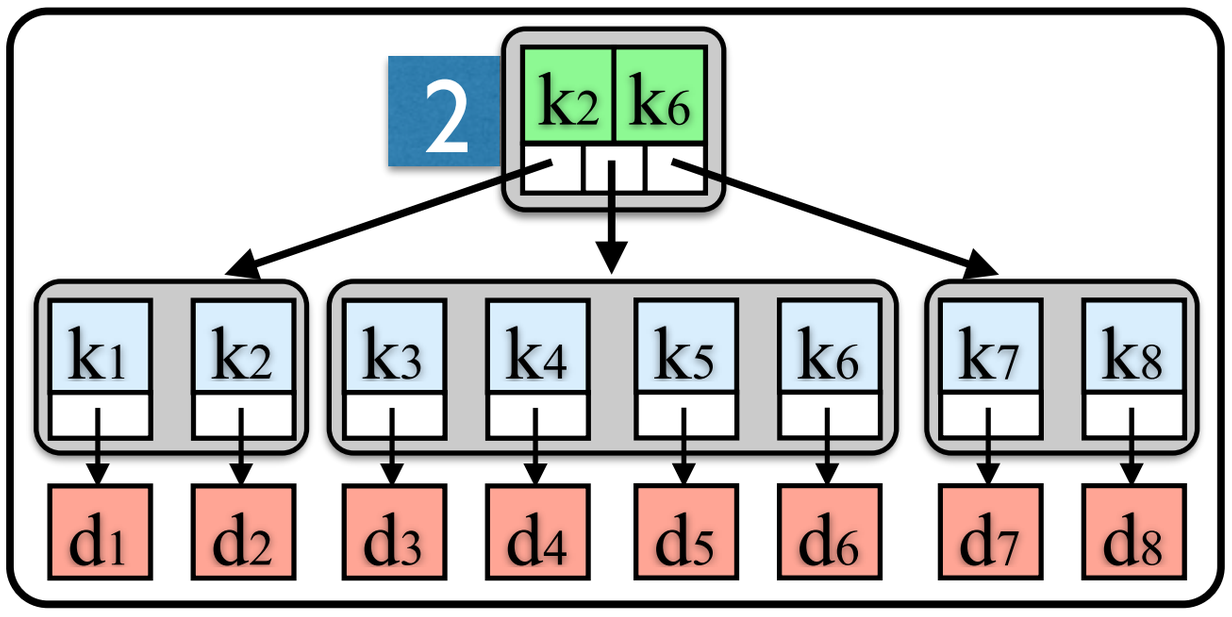
$I \cup I_{deg}$

- Re-degreeing extension affects ALL proofs
  - Have to redo all proofs, even those of list-only operations

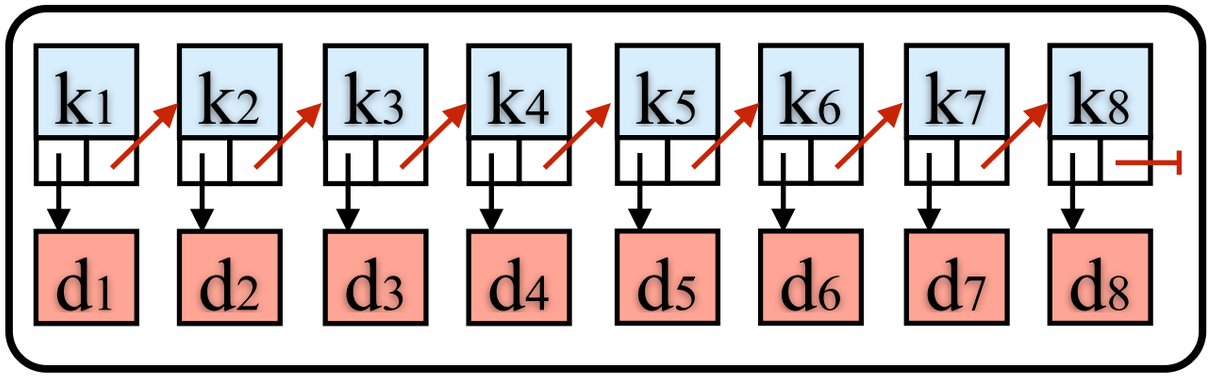
B+ Tree <K,V>: Possible Extension (CoLoSL)



$I \cup I_{deg}$

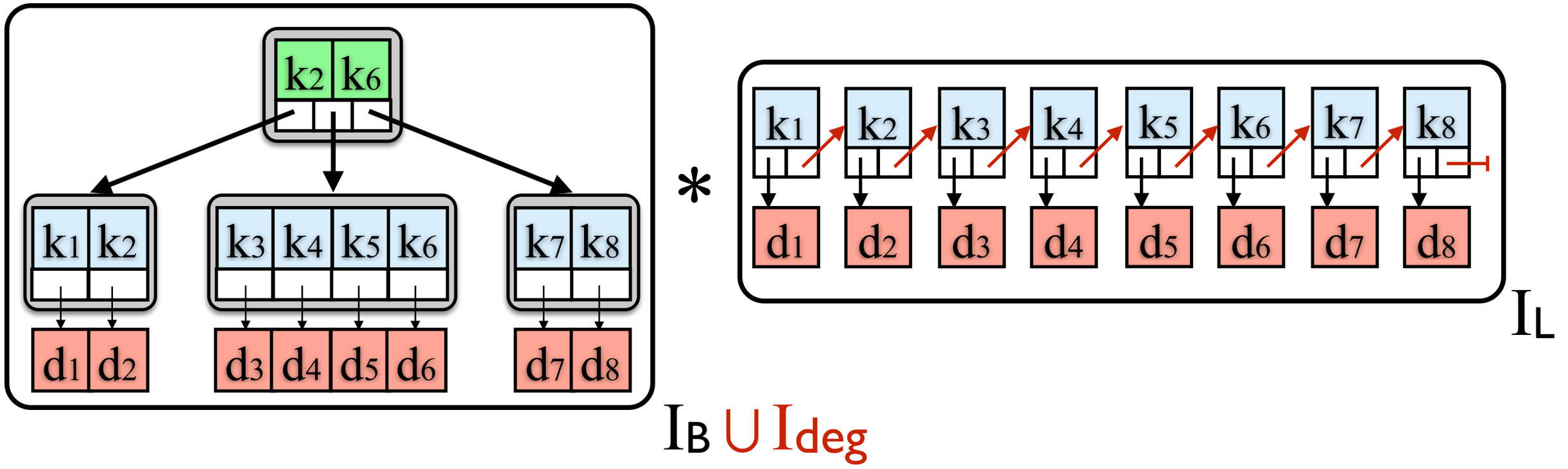


$I_B \cup I_{deg}$



I_L

B+ Tree <K,V>: Possible Extension (CoLoSL)



- Re-degreeing extension only affects the tree
- Re-degreeing extension does NOT affect the list proofs
  - Can reuse the proofs as before

Conclusions

- From OG/Rg to CAP/TaDA
  - Huge steps towards compositionality/locality
  - Are we there yet? No!
- CoLoSL
  - Subjective/overlapping views
  - Dynamic framing on shared resource/interference
  - Dynamic extension
  - Are we there yet? Still No!
    - Abstraction layers; abstract atomicity, …

Questions?

Thank you for listening