CoLoSL

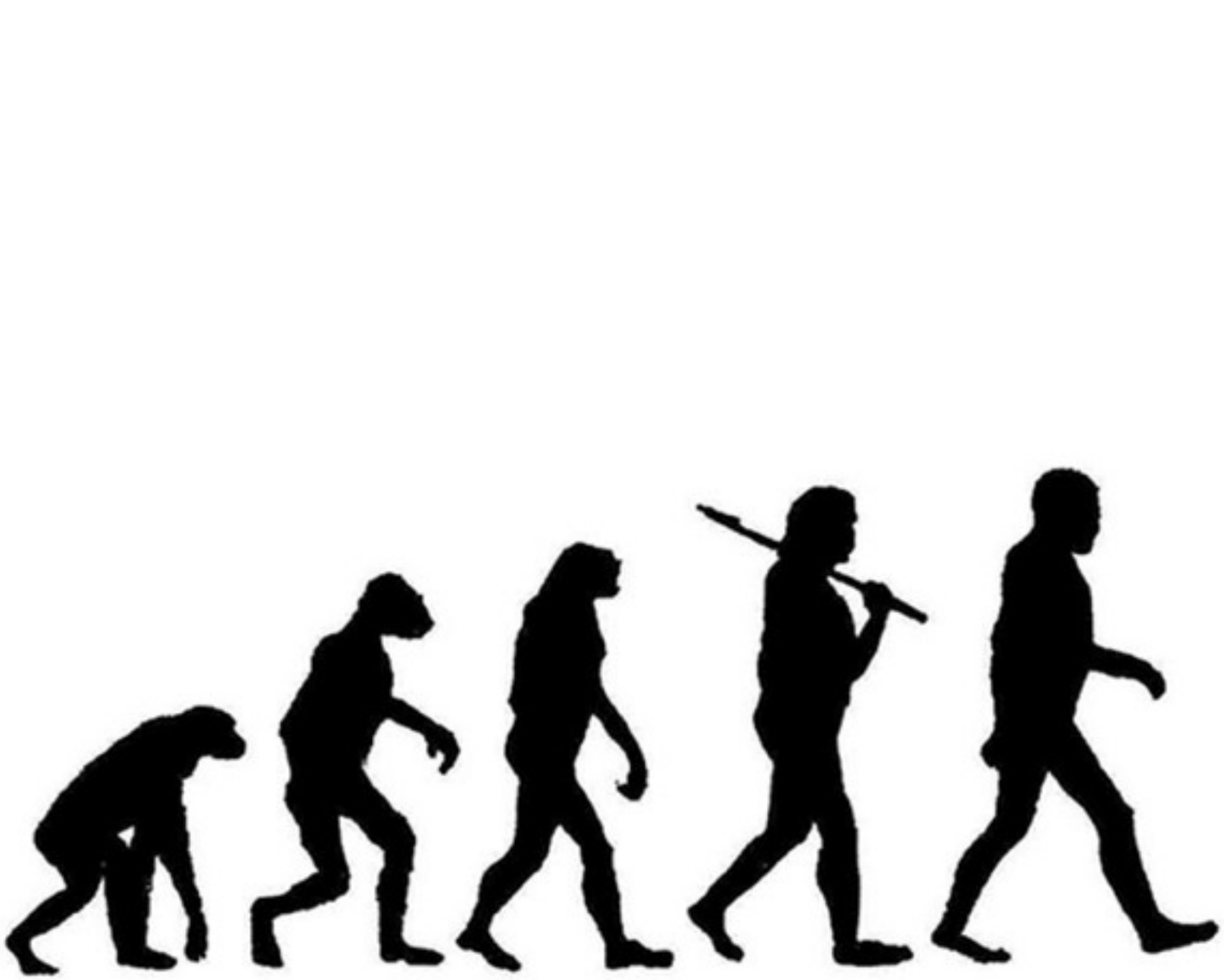
Concurrent Local Subjective Logic

Azalea Raad Jules Villard Philippa Gardner

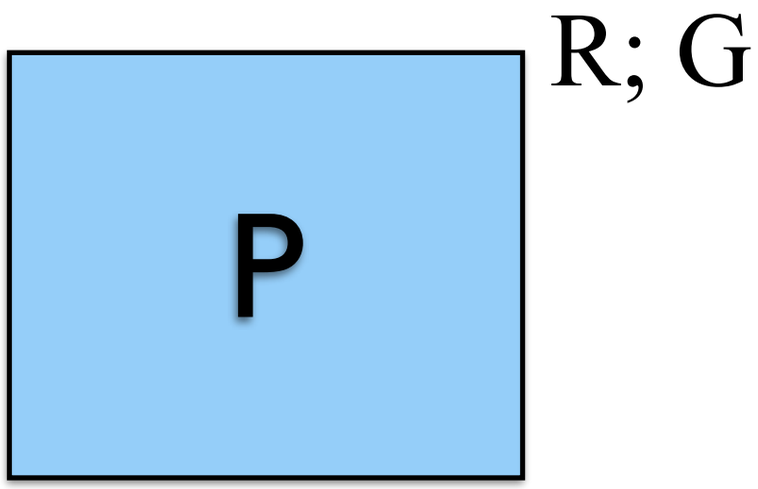
Imperial College London

7 May 2015

One Logic to Rule Them All…



Global Reasoning



$$\frac{\{\boxed{P}^{R;\,G}\}\ C1\ \{\boxed{Q1}^{R;\,G}\} \qquad \{\boxed{P}^{R;\,G}\}\ C2\ \{\boxed{Q2}^{R;\,G}\}}{\{\boxed{P}^{R;\,G}\}\ C1 \parallel C2\ \{\boxed{Q1 \wedge Q2}^{R;\,G}\}}$$

Rely-Guarantee (Owicki-Gries)

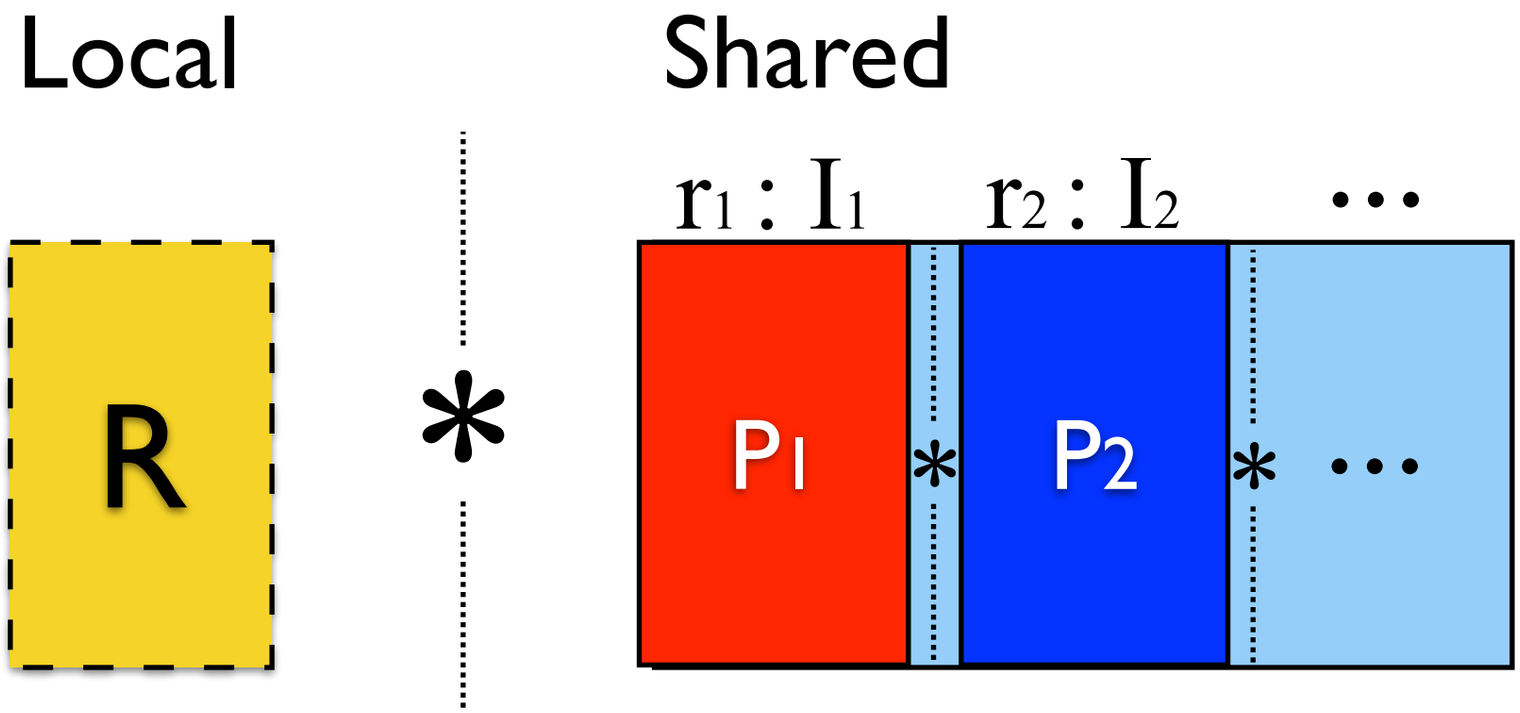
Global Reasoning

$$\frac{\{\boxed{P}^{R;\,G}\}\ C1\ \{\boxed{Q1}^{R;\,G}\} \qquad \{\boxed{P}^{R;\,G}\}\ C2\ \{\boxed{Q2}^{R;\,G}\}}{\{\boxed{P}^{R;\,G}\}\ C1 \parallel C2\ \{\boxed{Q1 \wedge Q2}^{R;\,G}\}}$$

- **No framing on shared resources / interference**
  - Reasoning on GLOBAL resources
  - Interference on ALL resources considered

Rely-Guarantee (Owicki-Gries)

Local Reasoning (Disjoint)



$$\frac{\left\{\boxed{P}^{r_1}_{I_1}\right\} \;C\; \left\{\boxed{P'}^{r_1}_{I_1}\right\}}{\left\{\boxed{P}^{r_1}_{I_1} * \boxed{Q}^{r_2}_{I_2}\right\} \;C\; \left\{\boxed{P'}^{r_1}_{I_1} * \boxed{Q}^{r_2}_{I_2}\right\}}\ (\textsc{Frame})$$

CSL, RGSep, DG, CAP, HOCAP, iCAP, FCSL, TaDA

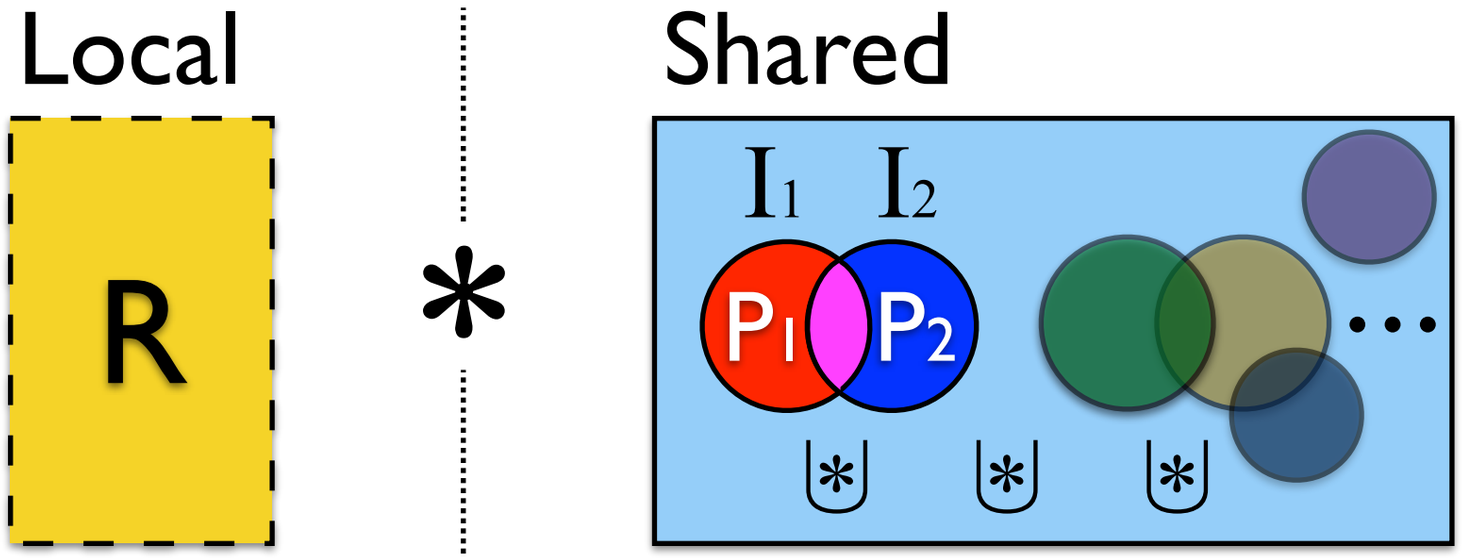
Local Reasoning (Disjoint)

$$\frac{\left\{ \boxed{P}^{r_1}_{I1} \right\} \; C \; \left\{ \boxed{P'}^{r_1}_{I1} \right\}}{\left\{ \boxed{P}^{r_1}_{I1} * \boxed{Q}^{r_2}_{I2} \right\} \; C \; \left\{ \boxed{P'}^{r_1}_{I1} * \boxed{Q}^{r_2}_{I2} \right\}} \text{(FRAME)}$$

- **<u>Limited framing on shared resources / interference</u>**
  - **Static** (pre-determined) frames (regions/ invariants)
  - **Physically** disjoint frames

CSL, RGSep, DG, CAP, HOCAP, iCAP, FCSL, TaDA

CoLoSL: Concurrent Local Subjective Logic



$$\frac{\{\boxed{P}_I\}\ C\ \{\boxed{P'}_I\} \qquad I \cup I' \sqsubseteq^{P} I}{\{\boxed{P \circledast Q}_{I \cup I'}\}\ C\ \{\boxed{P' \circledast Q}_{I \cup I'}\}}\ (\textsc{Frame})$$

CoLoSL

CoLoSL: Concurrent Local Subjective Logic

$$\frac{\left\{\boxed{P}_{I}\right\}\ C\ \left\{\boxed{P'}_{I}\right\} \qquad I \cup I' \sqsubseteq^{P} I}{\left\{\boxed{P \circledast Q}_{I \cup I'}\right\}\ C\ \left\{\boxed{P' \circledast Q}_{I \cup I'}\right\}}\ (\textsc{Frame})$$

- **Flexible framing on shared resources/invariants**
  - **Overlapping** frames
  - **Flexible** framing/rewriting of interference

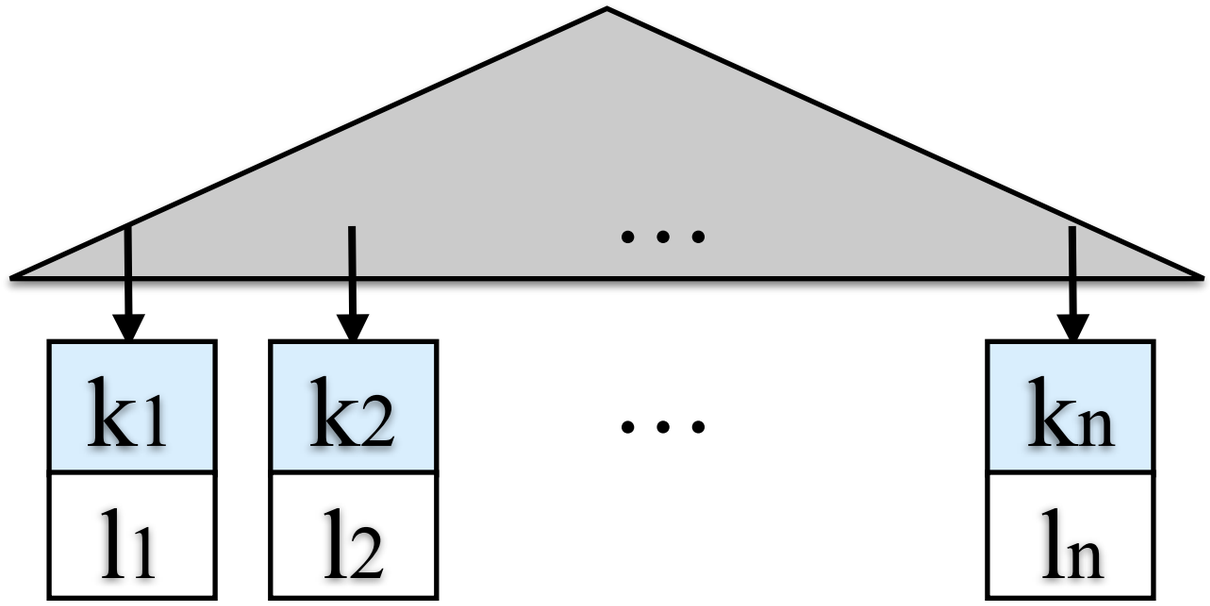
Local Rely-Guarantee

$$\frac{J \vdash \left\{ \boxed{P}_I \right\} C \left\{ \boxed{P'}_I \right\} \quad \text{precise}(J') \quad J' \triangleright (Q, I')}{J * J' \vdash \left\{ \boxed{P * Q}_{I * I'} \right\} C \left\{ \boxed{P' * Q}_{I * I'} \right\}} \text{(FRAME)}$$

$$\frac{R; G; J \vdash \{P\}\ C\ \{P'\} \quad \text{precise}(J') \quad J' \triangleright (Q, R', G')}{R*R'; G*G'; J*J' \vdash \{P * Q\}\ C\ \{P' * Q\}} \text{(FRAME)}$$

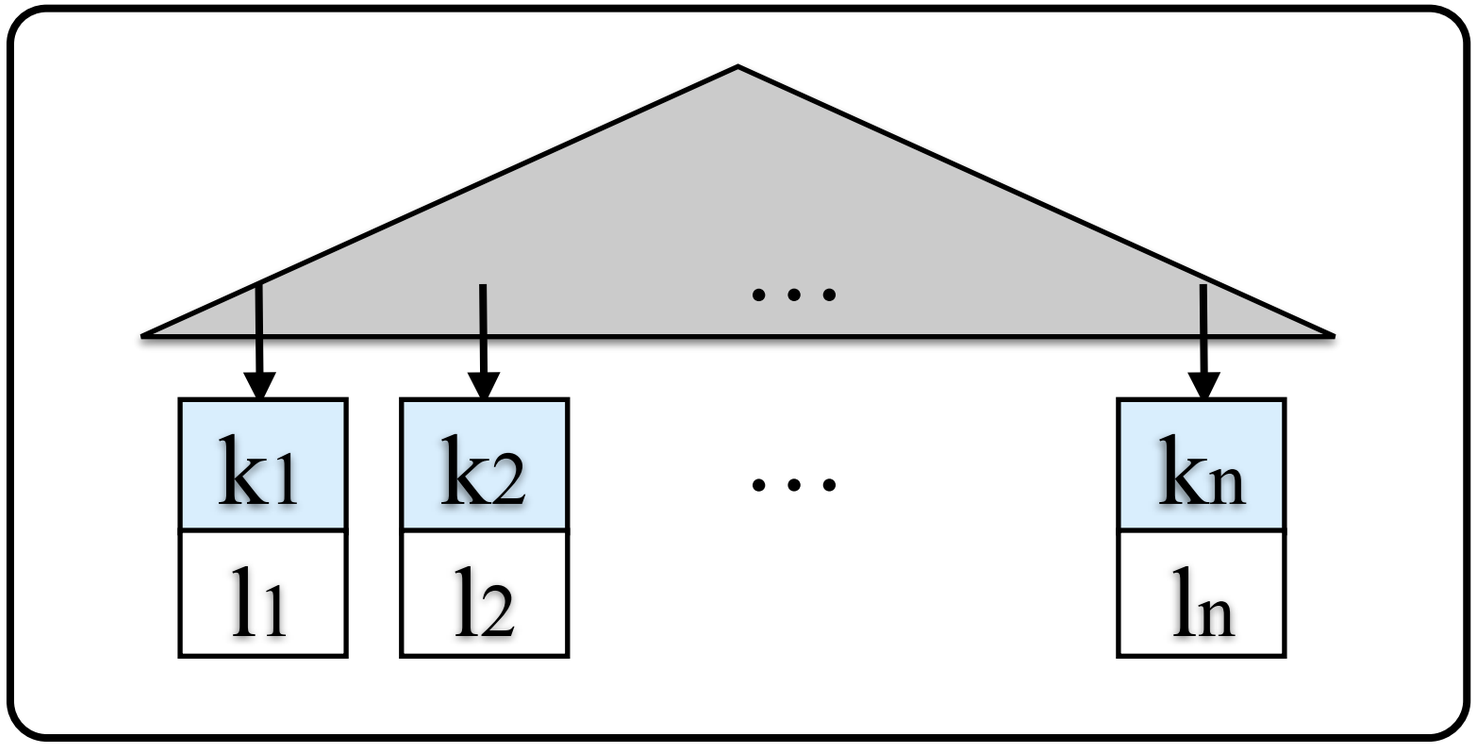
LRG

Concurrent Balanced Search Tree



- **Tree operations**
  - find_BS(k); add_BS(k, l); remove_BS(k)

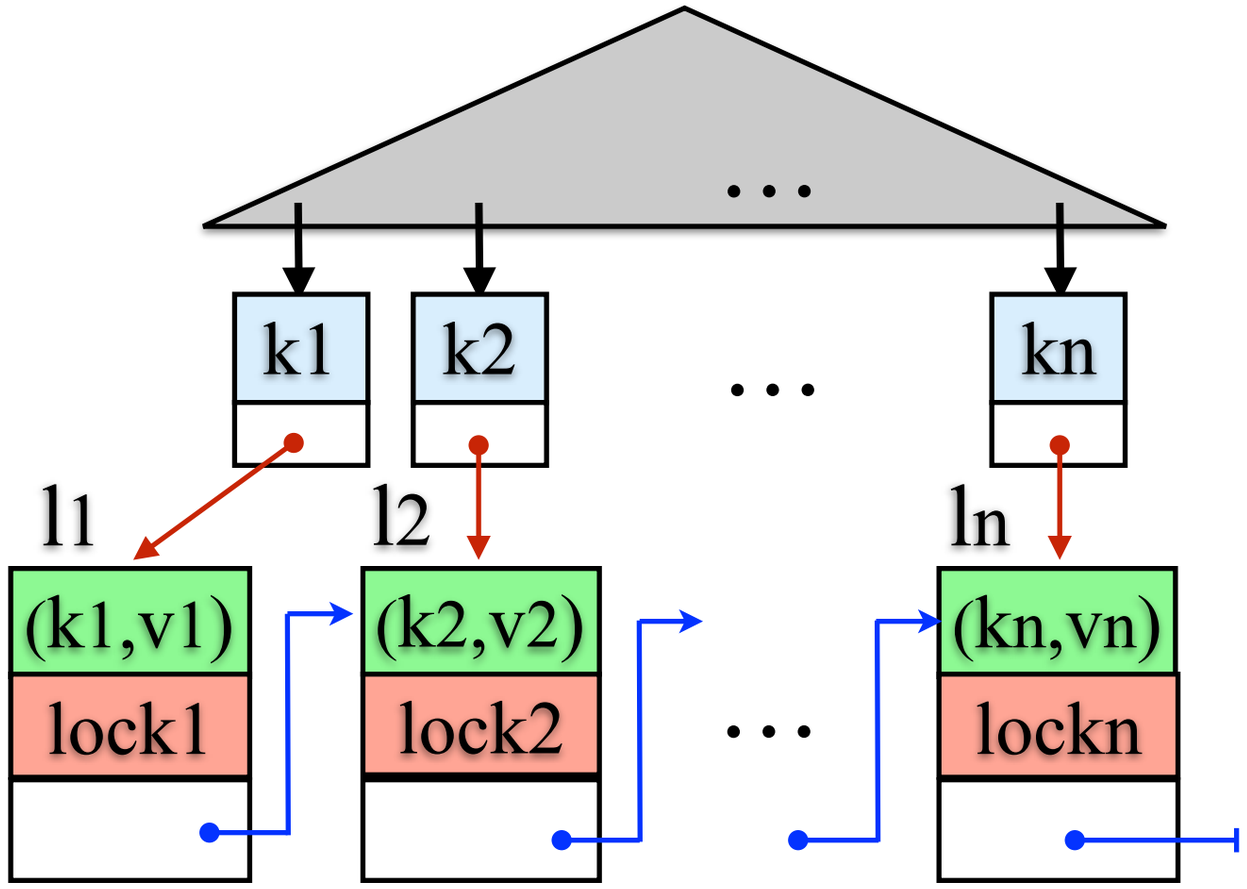
Concurrent Balanced Search Tree



$I_{BS} = I_{find} \cup I_{add} \cup I_{rem}$

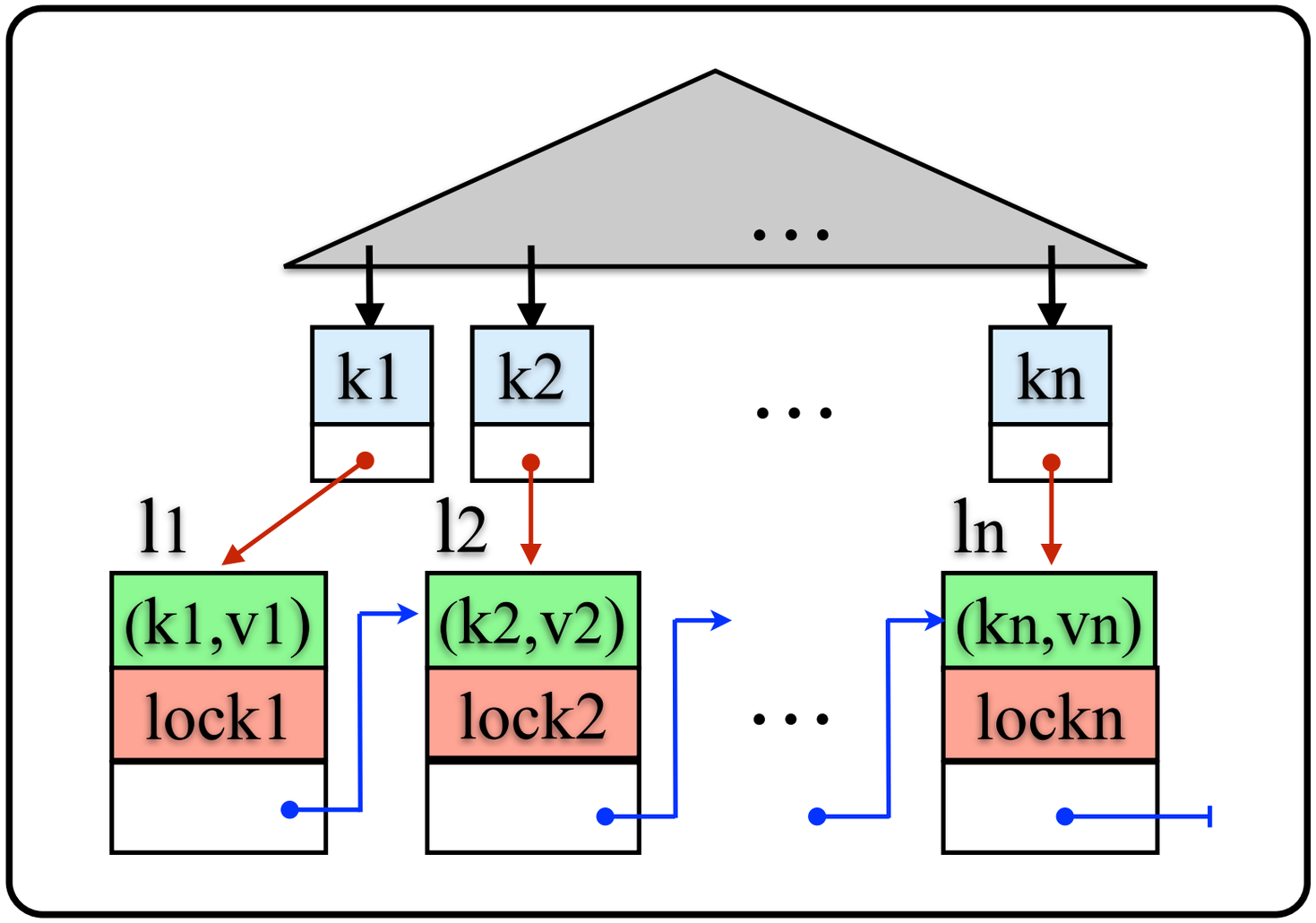
- **Tree operations**
  - find_BS(k); add_BS(k, l); remove_BS(k)

Concurrent B+ Tree



- **B+Tree operations**
  - find_B+(k); update(k, v); updateAll(V); add_B+(k,v); remove_B+(k)

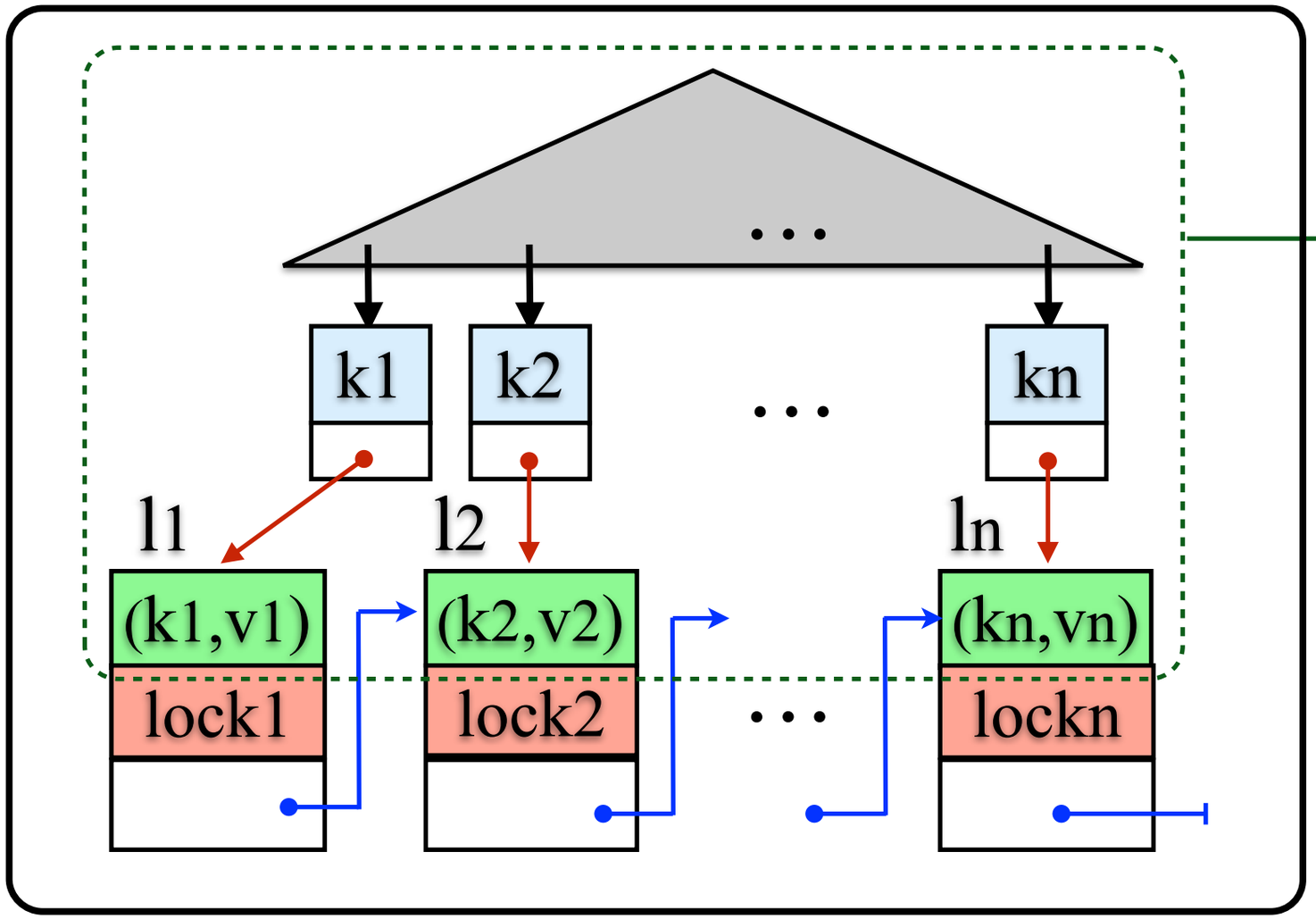
Concurrent B+ Tree



$$I_{B^+} = I_{BS} \cup I_{up} \cup I_{add\_L} \cup I_{rem\_L}$$

- **B+Tree operations**
  - find_B+(k); update(k, v); updateAll(V); add_B+(k,v); remove_B+(k)

Concurrent B+ Tree



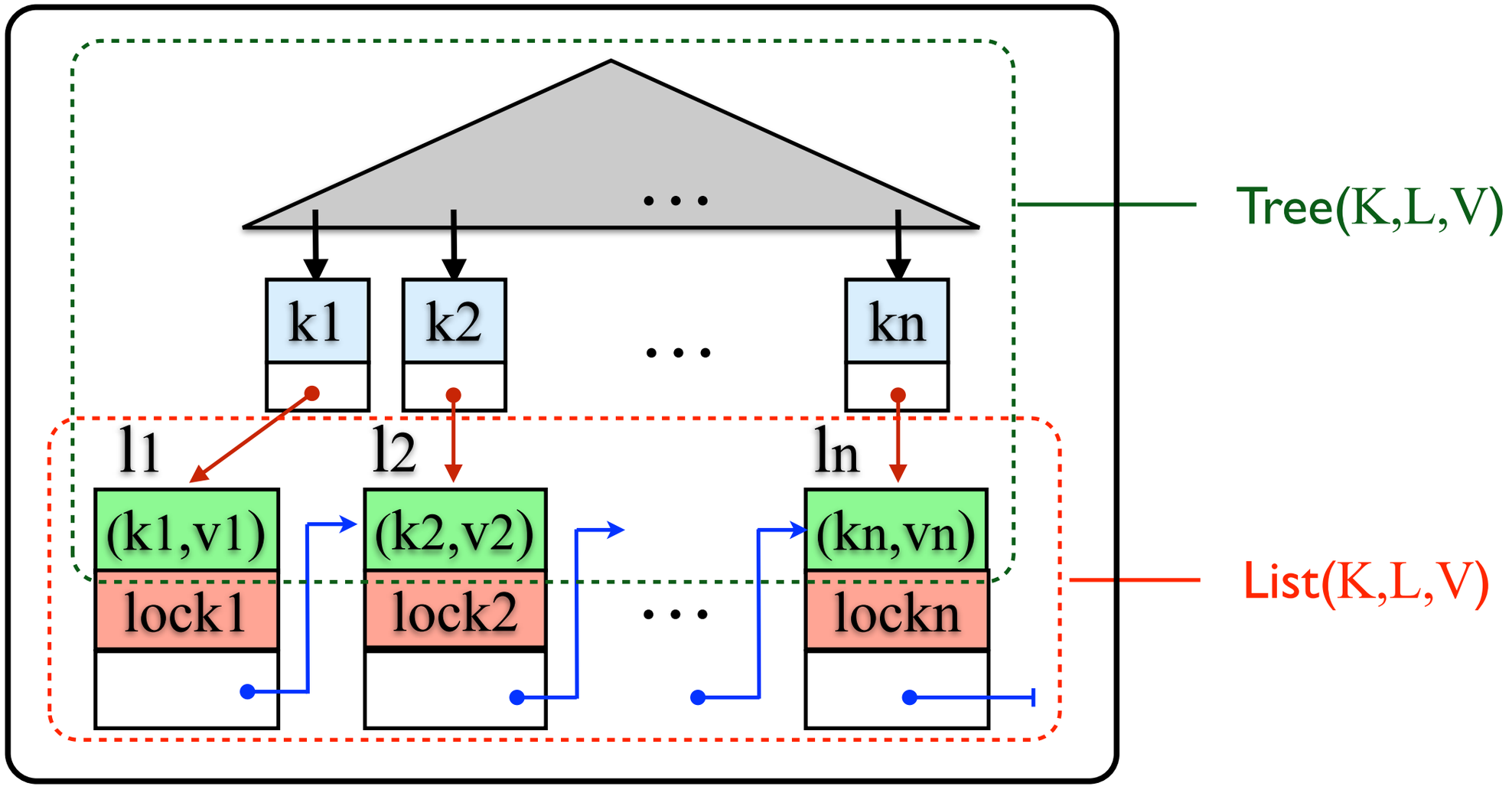
Tree(K,L,V)

$I_{B^+} = I_{BS} \cup I_{up} \cup I_{add\_L} \cup I_{rem\_L}$

- **B+Tree operations**
  - find_B+(k); update(k, v); updateAll(V); add_B+(k,v); remove_B+(k)

    $I_T = I_{BS} \cup I_{up}$

Concurrent B+ Tree



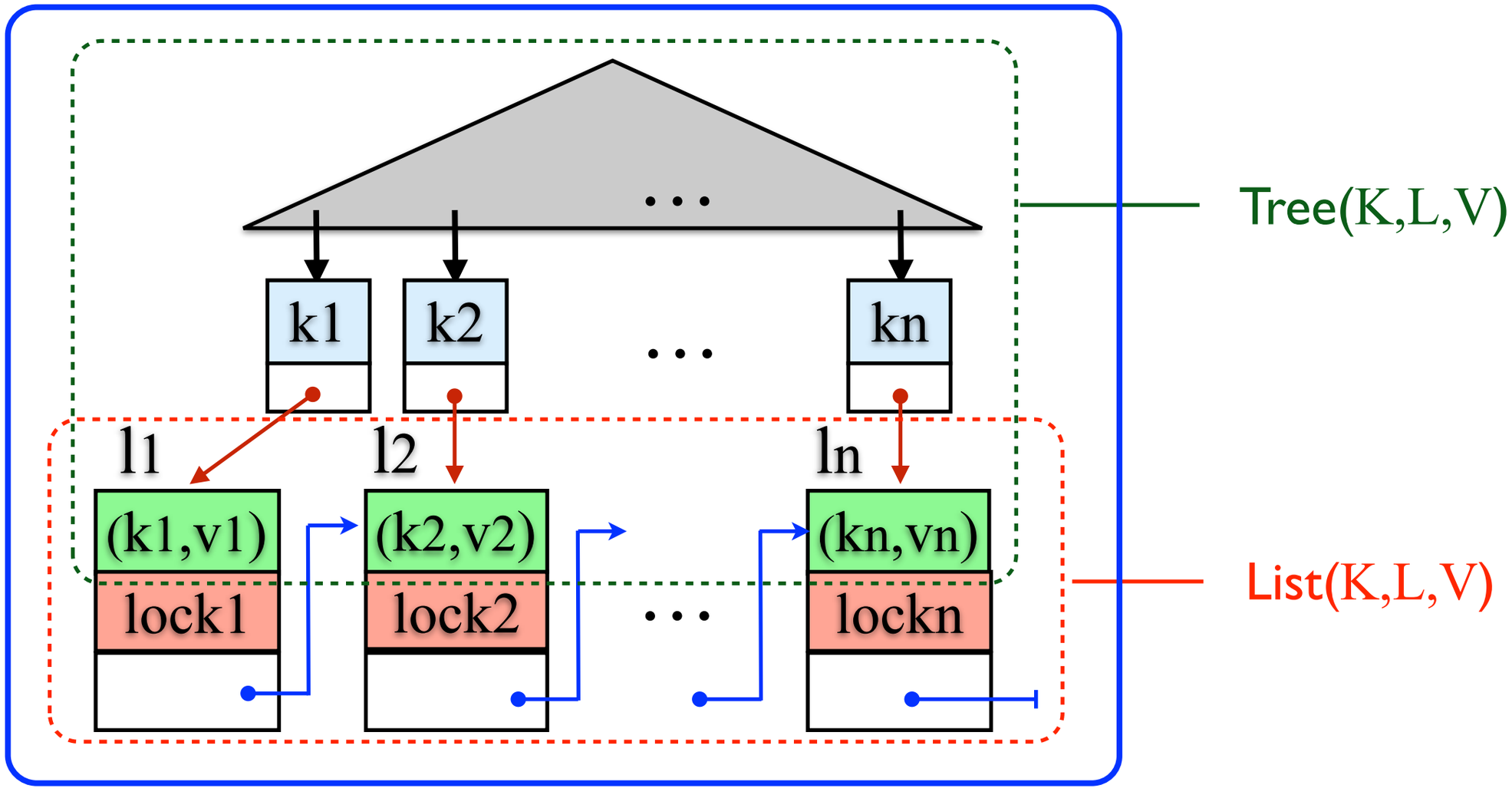
$I_{B^+} = I_{BS} \cup I_{up} \cup I_{add\_L} \cup I_{rem\_L}$

- **B+Tree operations**
  - find_B+(k); update(k, v); updateAll(V); add_B+(k,v); remove_B+(k)

$I_T = I_{BS} \cup I_{up}$ $I_L = I_{up} \cup I_{add\_L} \cup I_{rem\_L}$

||

Concurrent B+ Tree



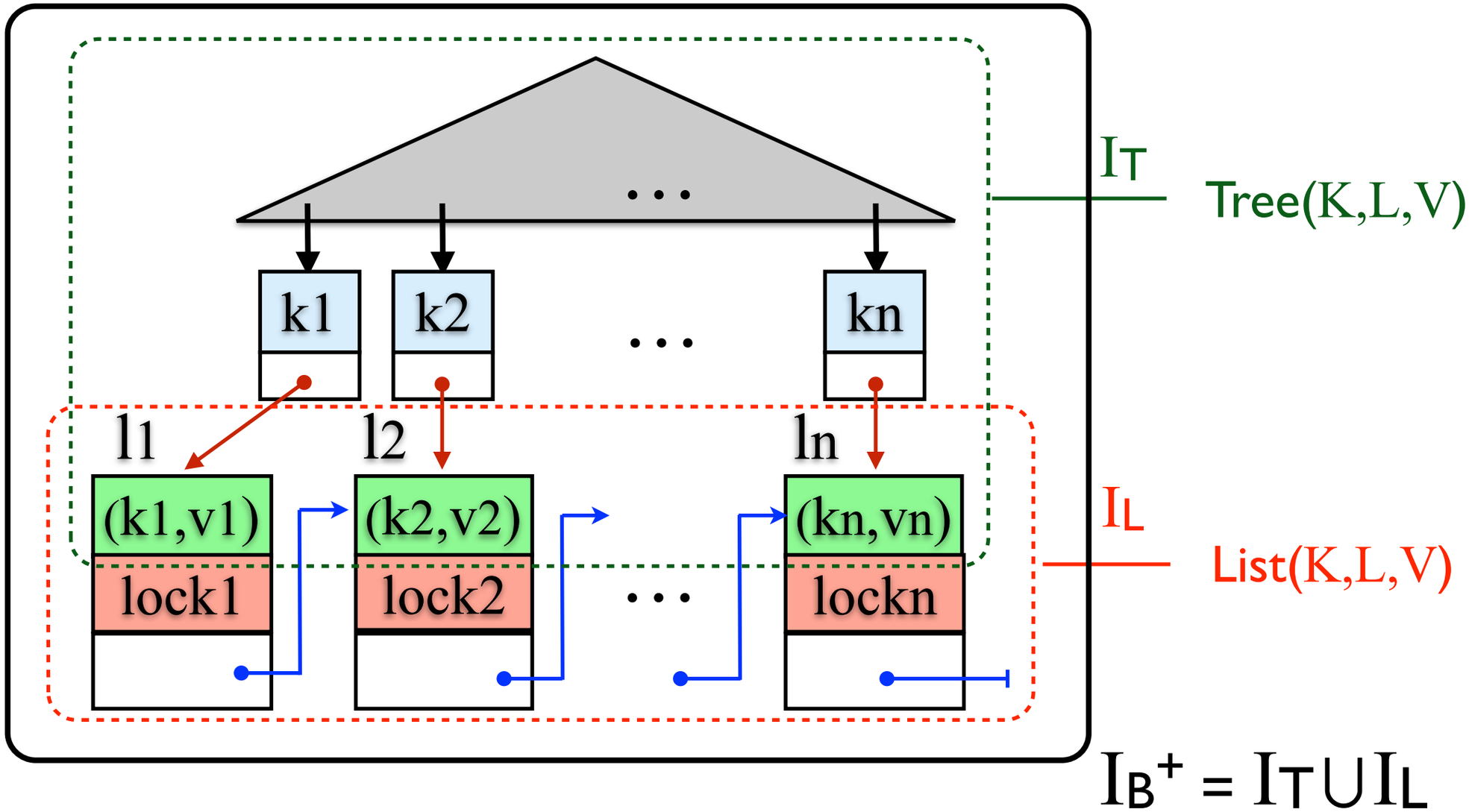
$I_{B^+} = I_{BS} \cup I_{up} \cup I_{add\_L} \cup I_{rem\_L}$

- **B+Tree operations**
  - find_B+(k); update(k, v); updateAll(V); add_B+(k,v); remove_B+(k)

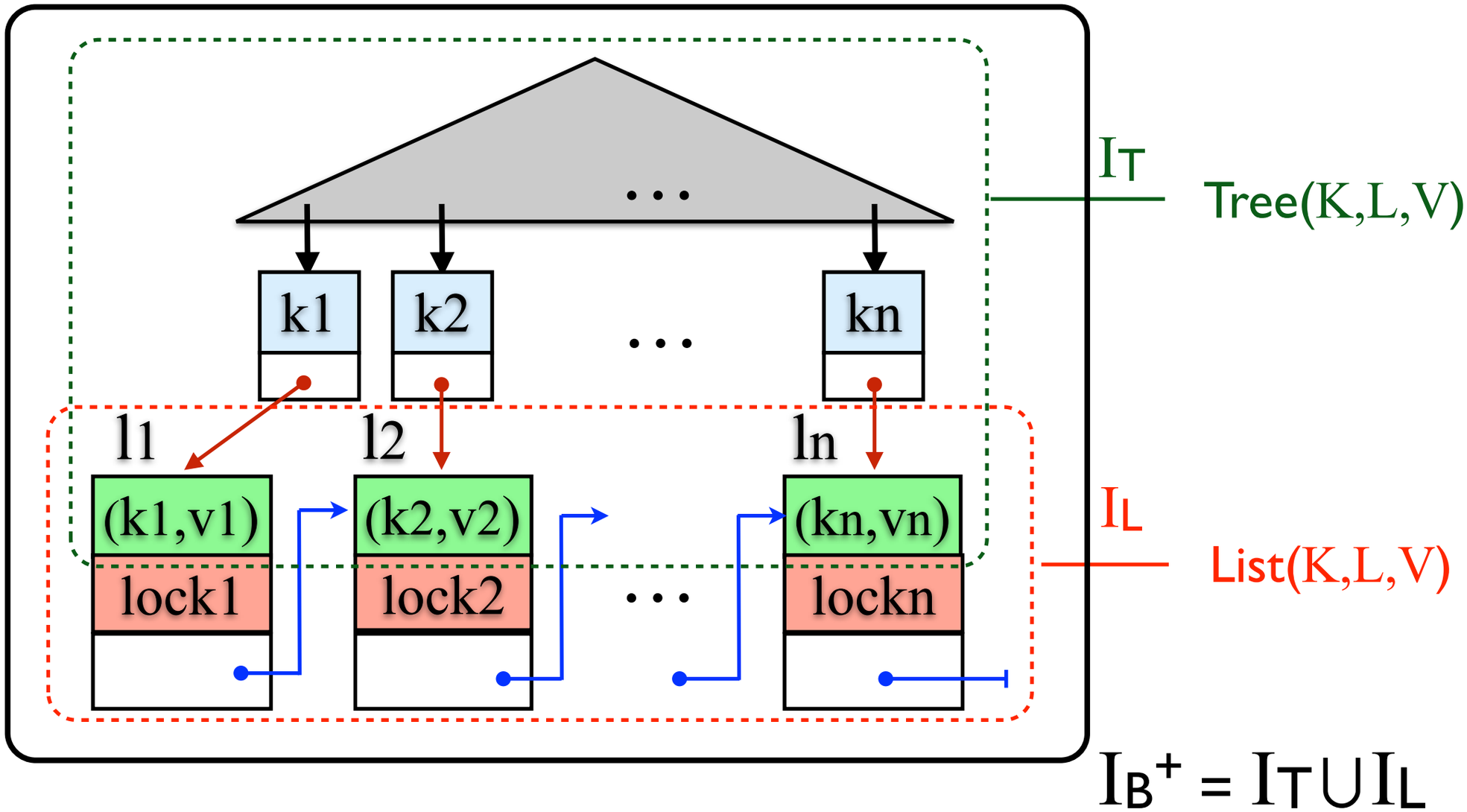
$I_T = I_{BS} \cup I_{up}$ $I_L = I_{up} \cup I_{add\_L} \cup I_{rem\_L}$ $I_{B^+} = I_T \cup I_L$

||

Concurrent B+ Tree

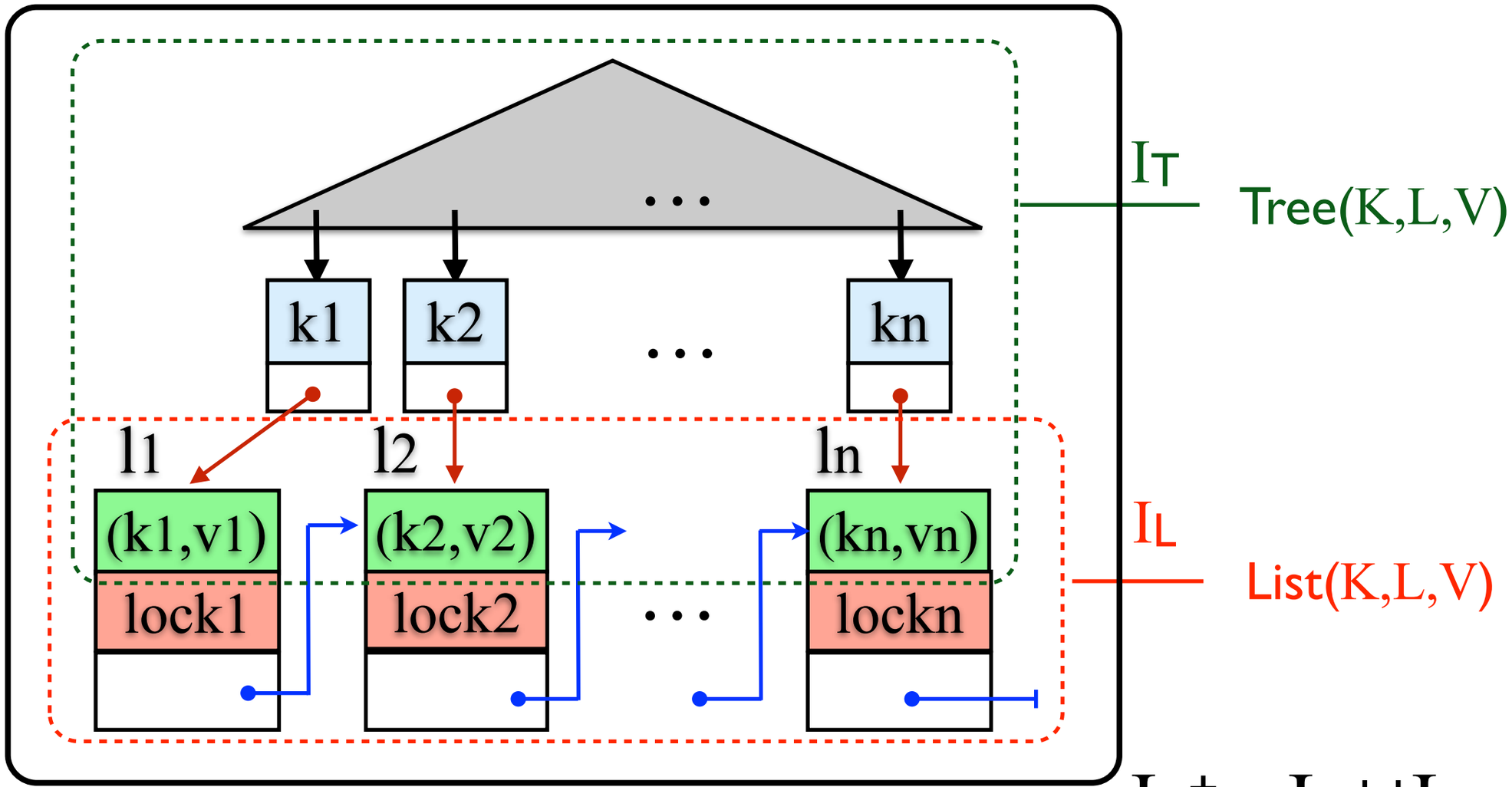


Concurrent B+ Tree



B+Tree(K, V) ⇔ ∃L. Tree(K,L,V) ⋈* List(K,L,V)

Concurrent B+ Tree

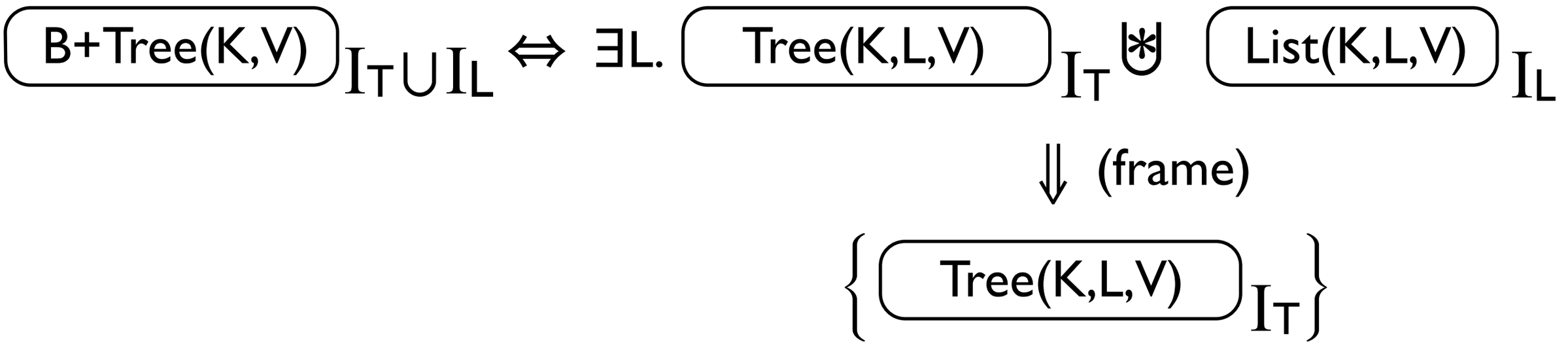


$I_{B^+} = I_T \cup I_L$

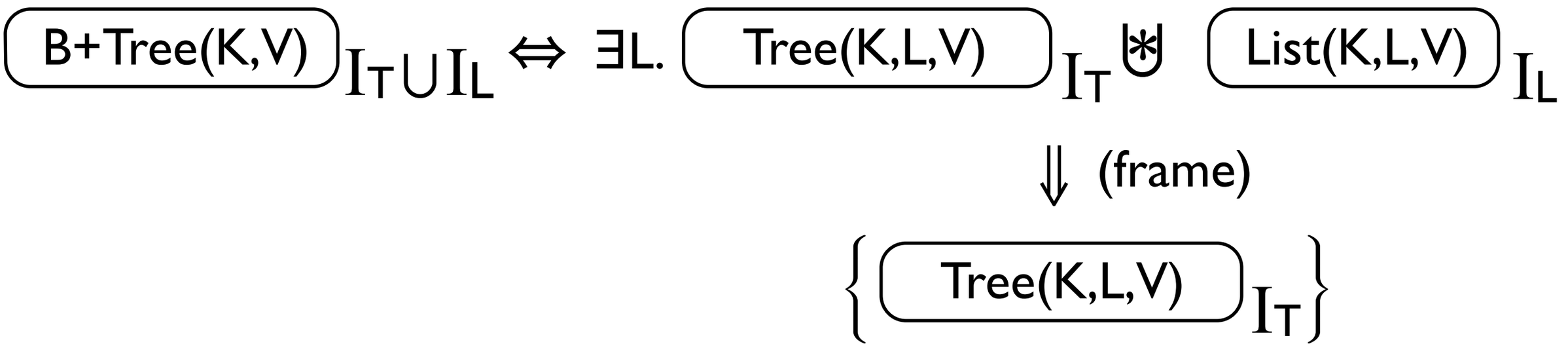
$$\boxed{\text{B+Tree(K,V)}}_{I_T \cup I_L} \Leftrightarrow \exists L.\ \boxed{\text{Tree(K,L,V)}}_{I_T} \circledast \boxed{\text{List(K,L,V)}}_{I_L}$$

Concurrent B+ Tree Wish List

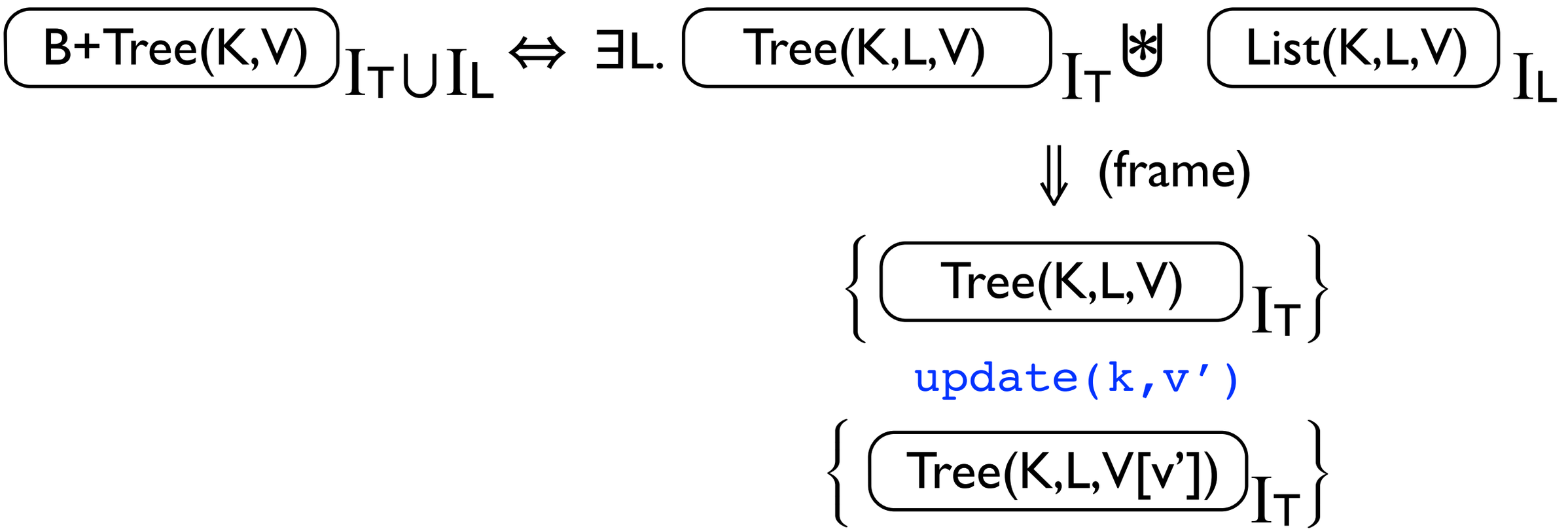
Concurrent B+ Tree Wish List



Concurrent B+ Tree Wish List

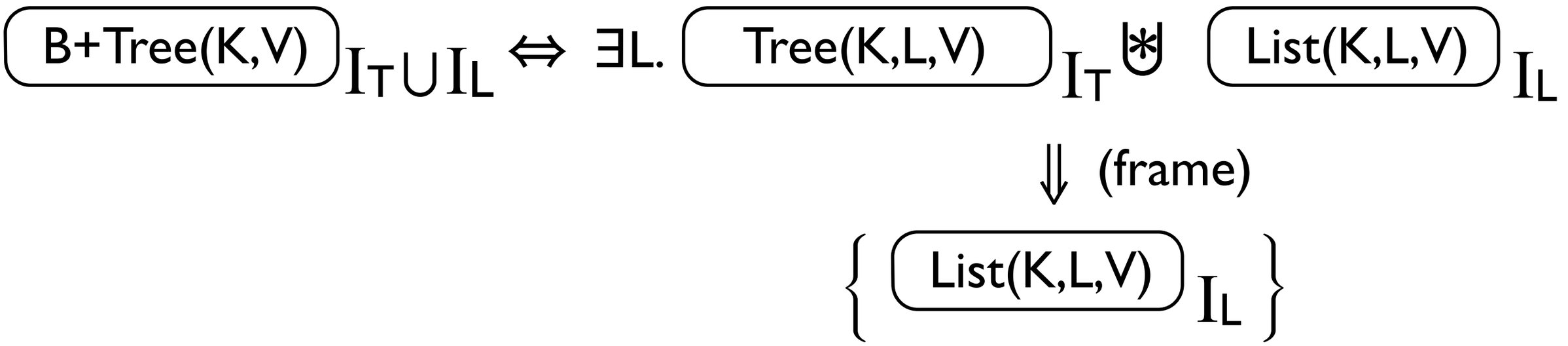


Concurrent B+ Tree Wish List

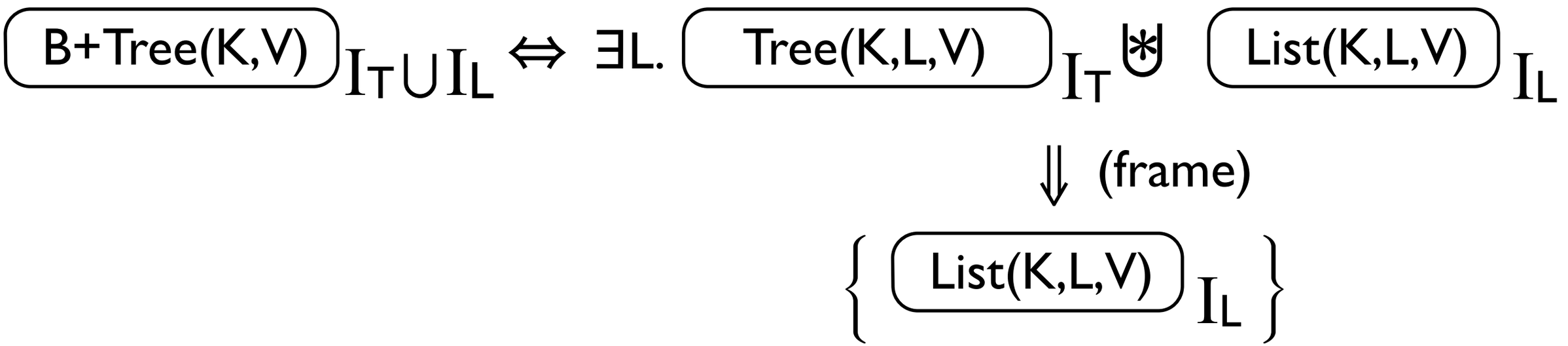


Concurrent B+ Tree Wish List

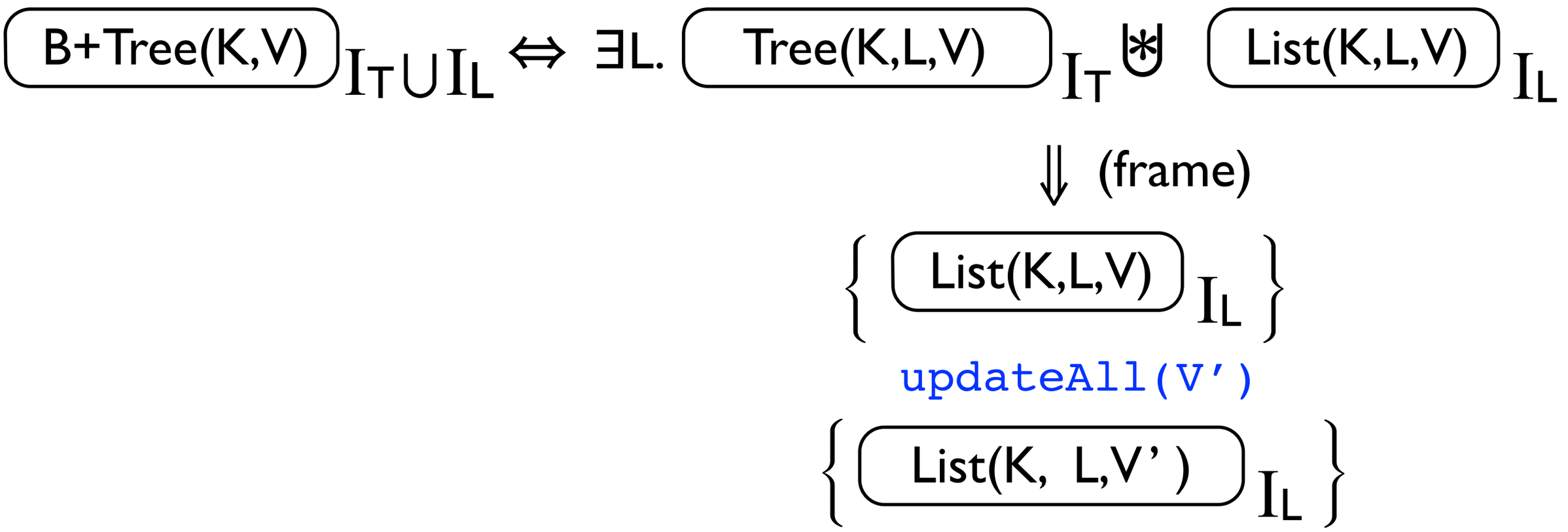
Concurrent B+ Tree Wish List



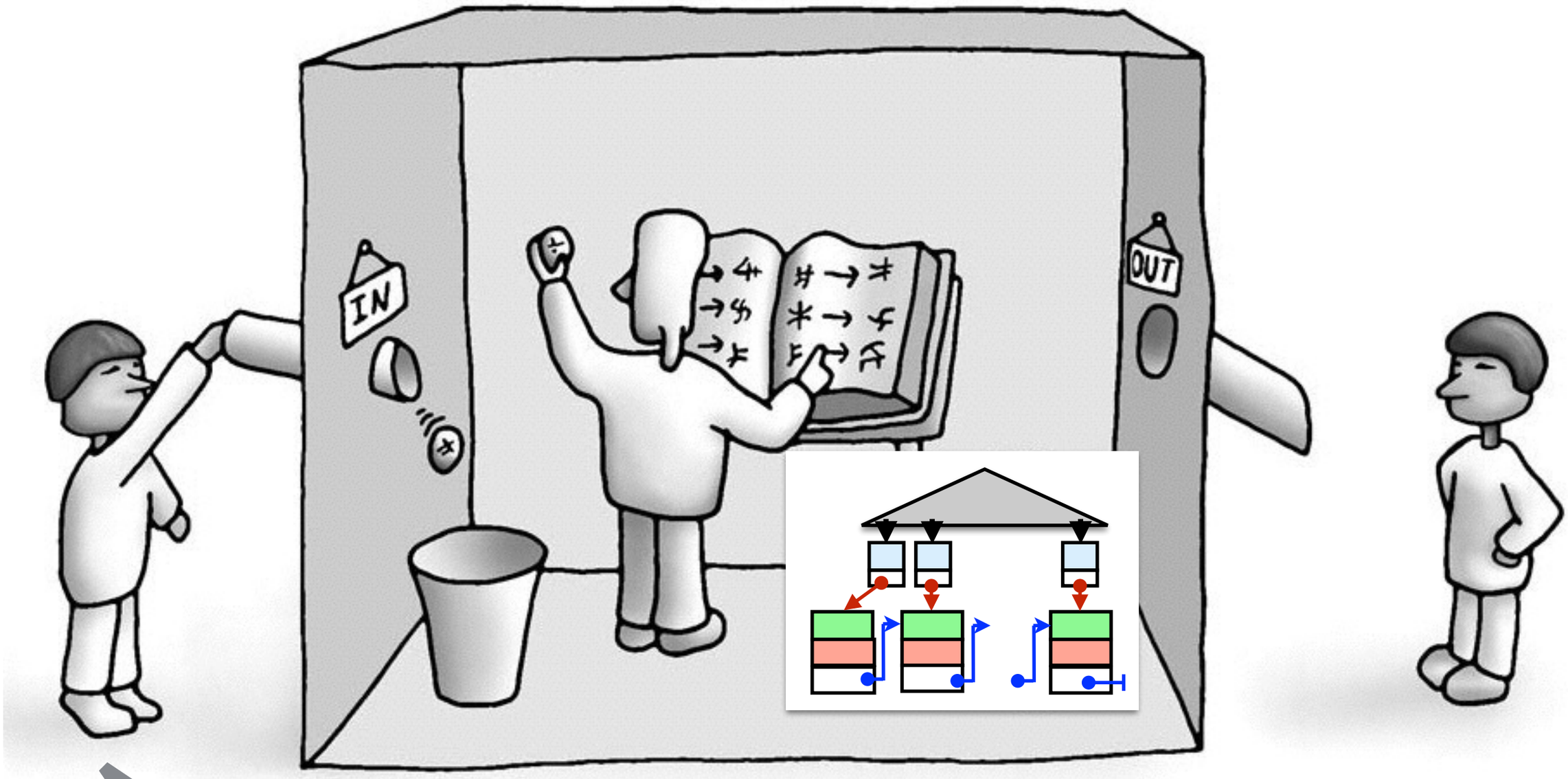
Concurrent B+ Tree Wish List



Concurrent B+ Tree Wish List

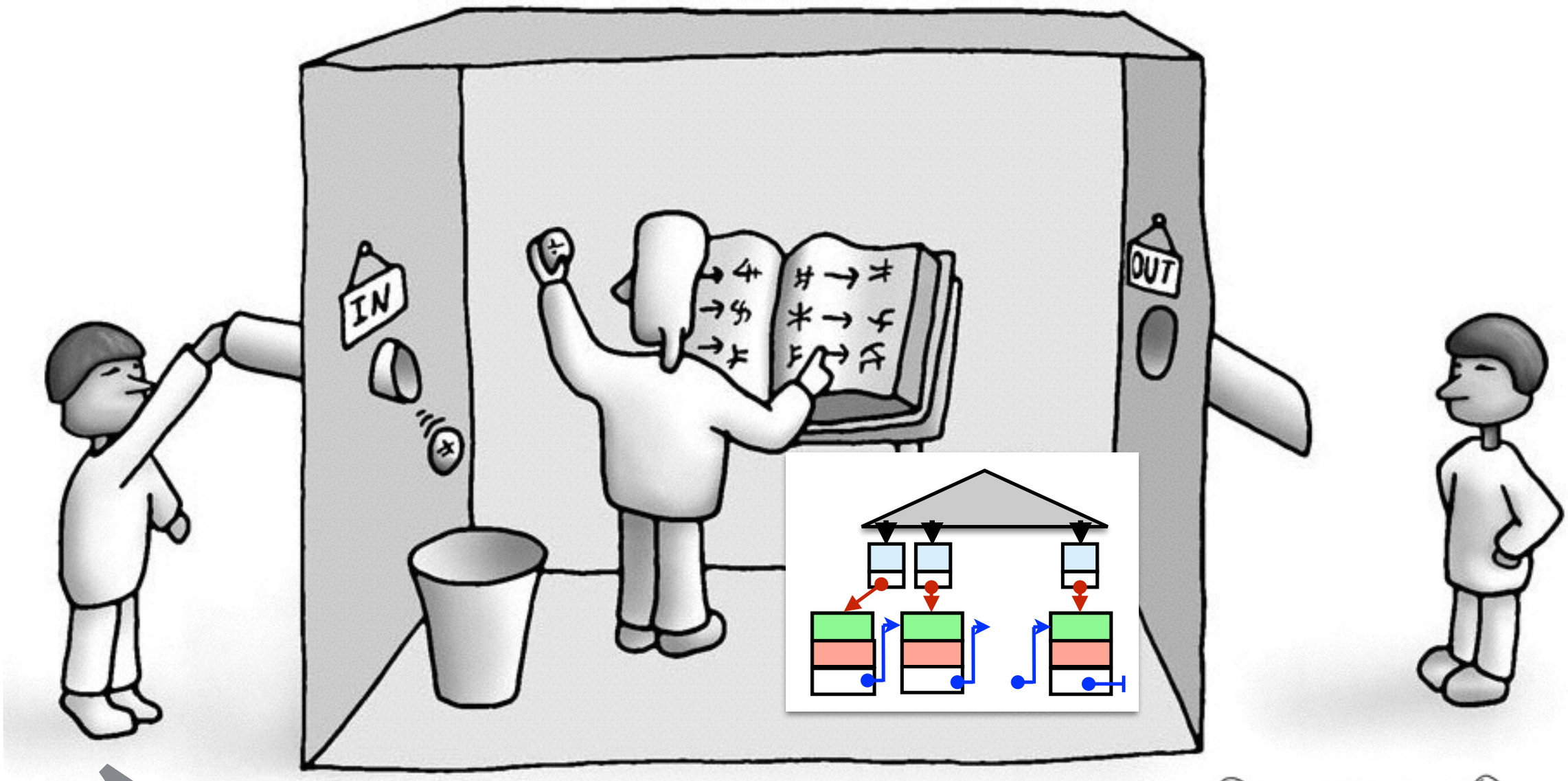


Chinese Room of Concurrent Modules



Are you a list?

Chinese Room of Concurrent Modules

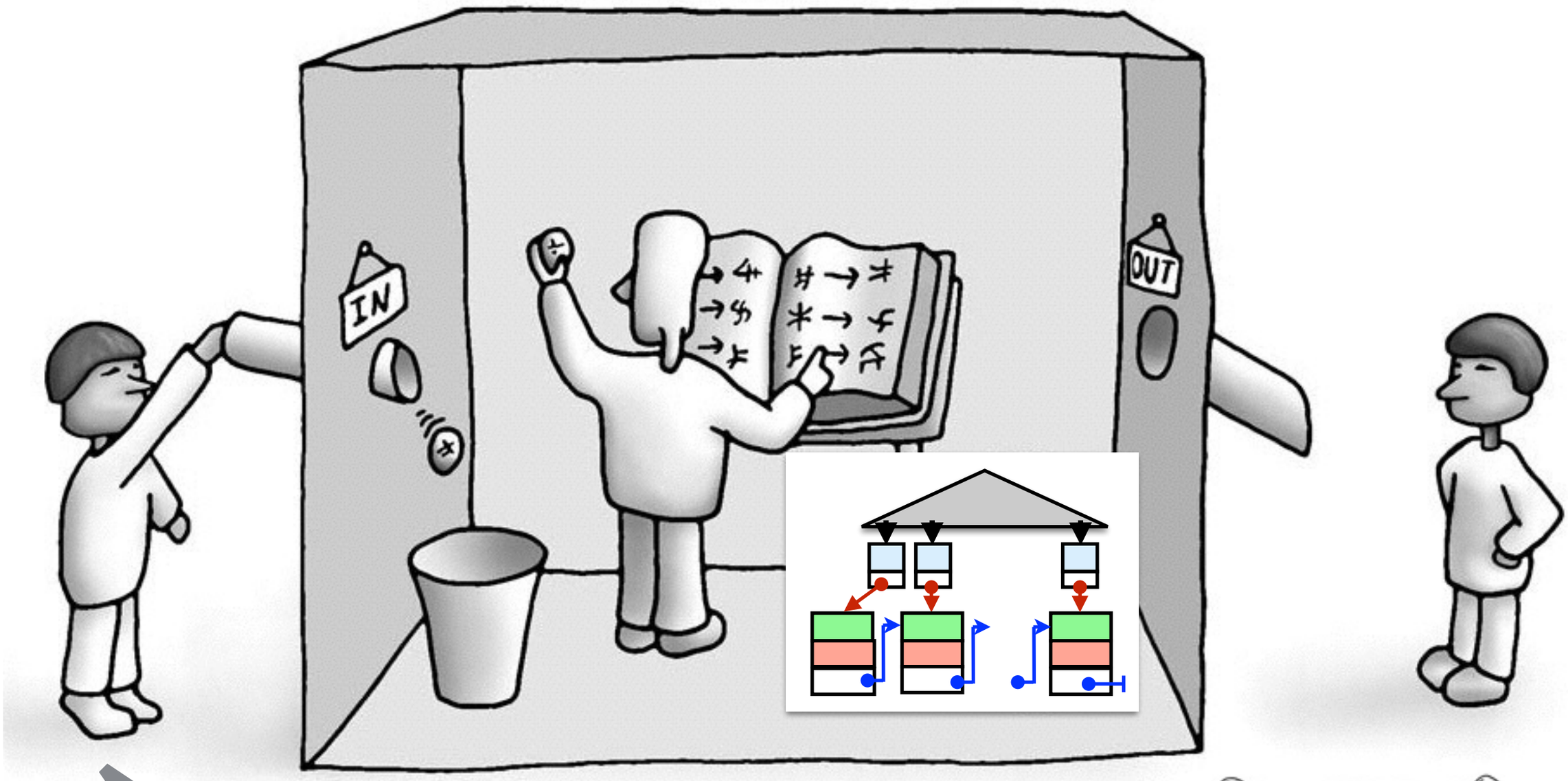


Are you a list?

Yes:

List(K,L,V)

Chinese Room of Concurrent Modules



Are you a tree?

Yes:

Tree(K,L,V)

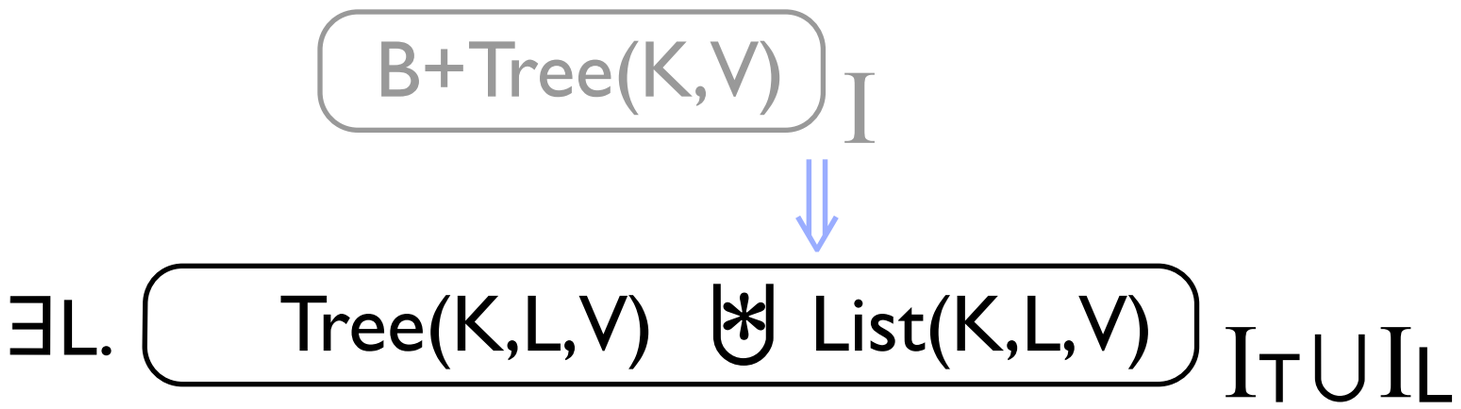
CoLoSL Principles

CoLoSL Principles

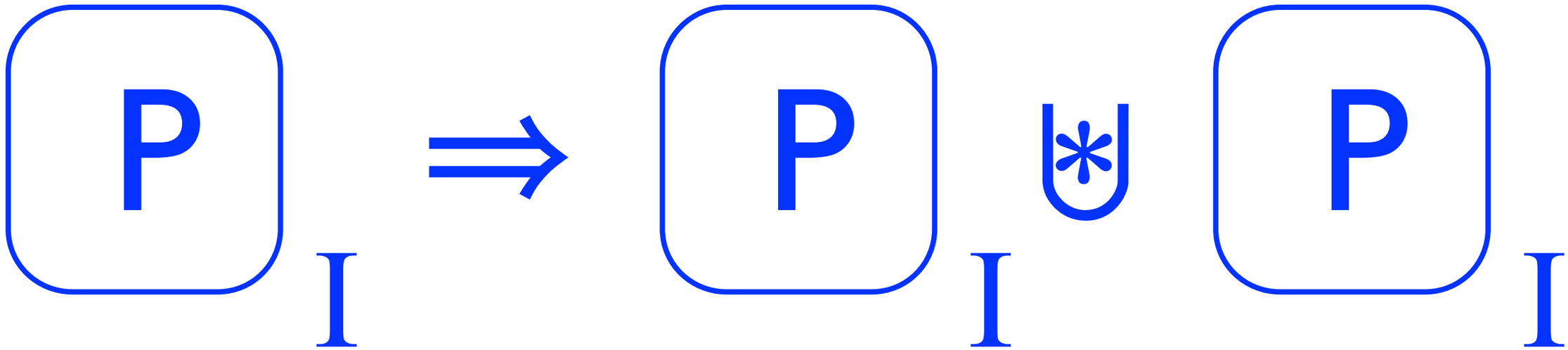
B+Tree(K,V) $_{I_T \cup I_L}$

$\exists L.$ Tree(K,L,V) $_{I_T}$ ⊛ List(K,L,V) $_{I_L}$

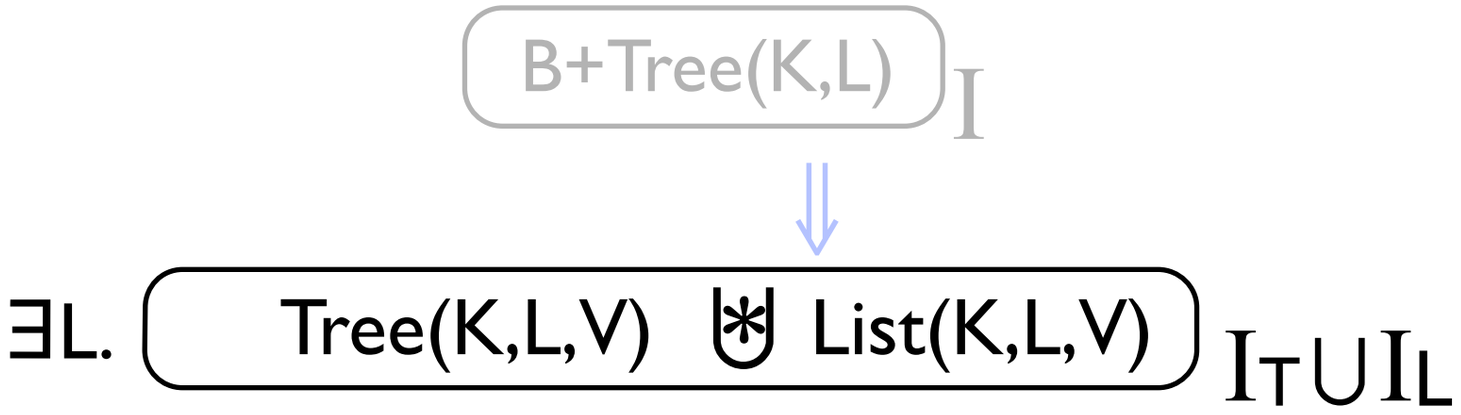
CoLoSL Principles



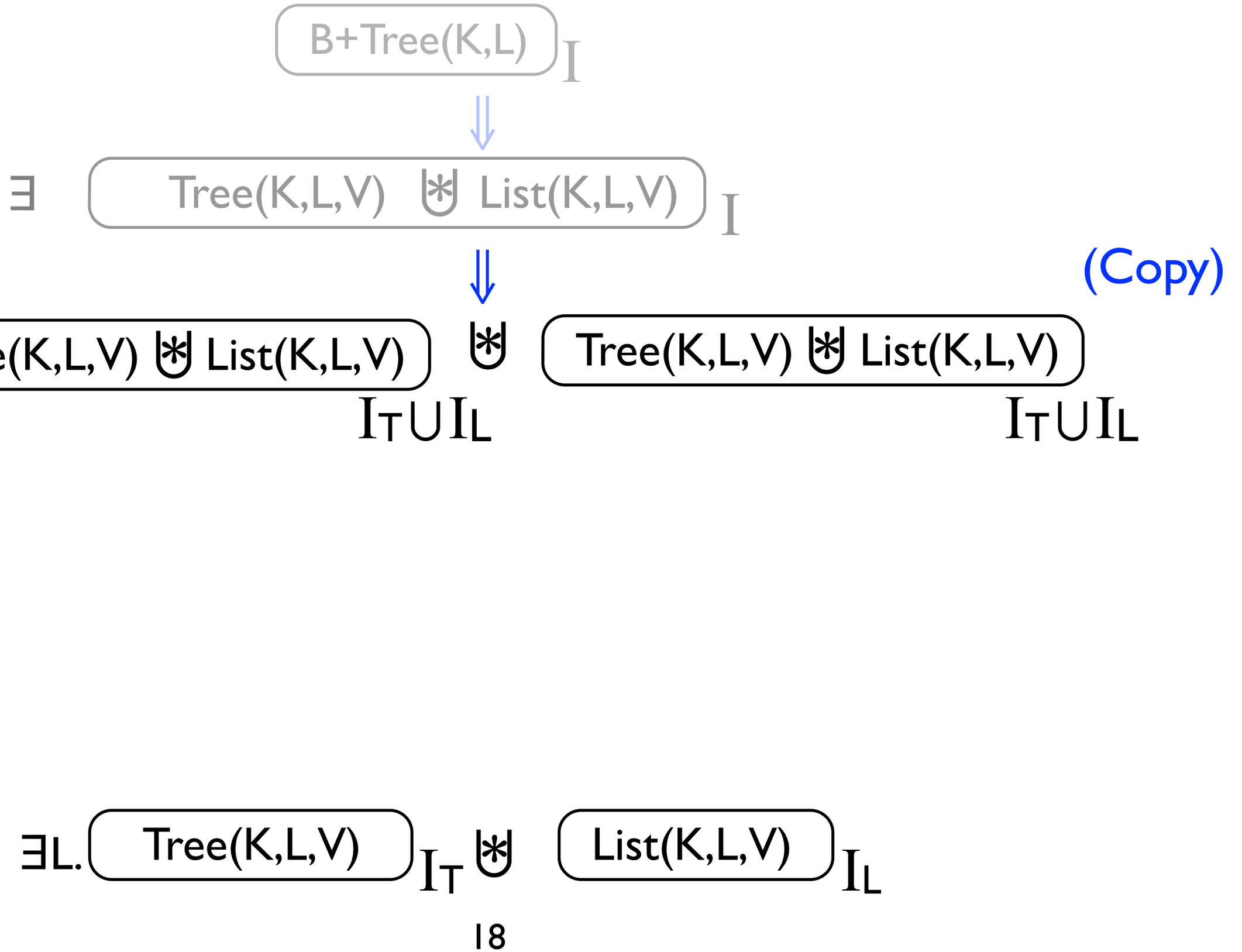
Duplicating Resources



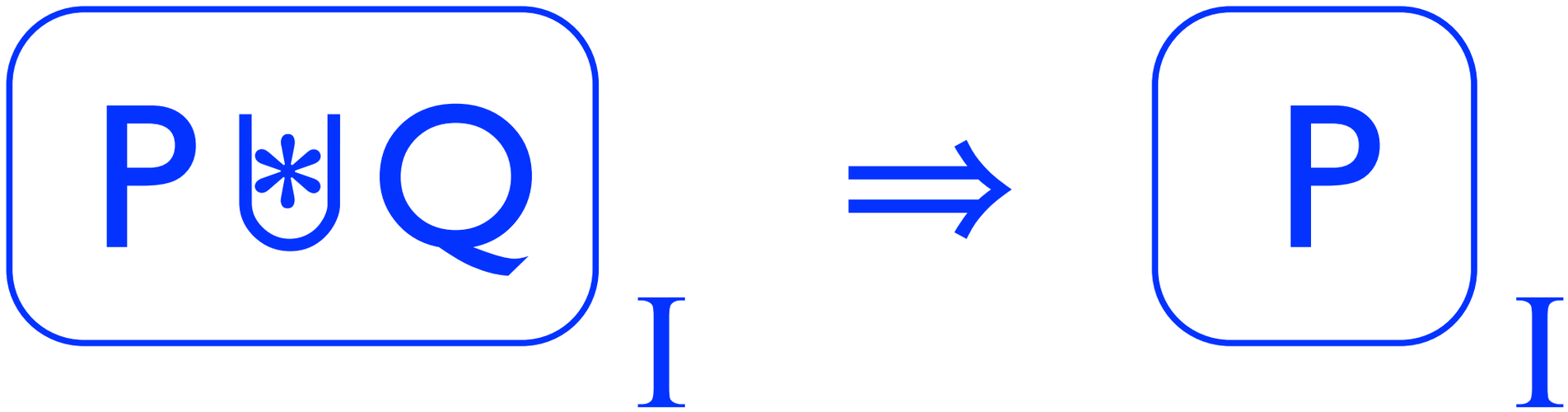
CoLoSL Principles



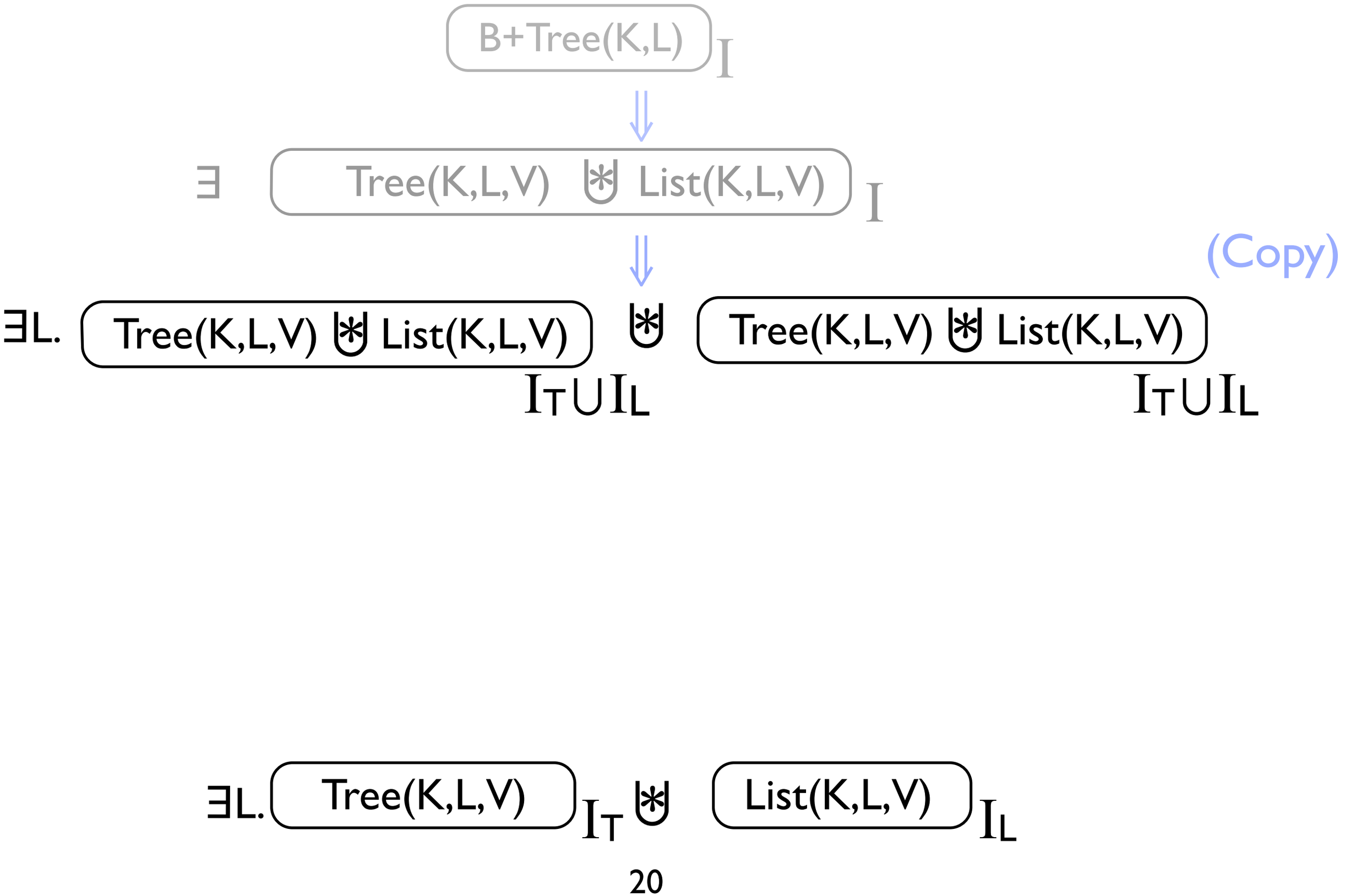
CoLoSL Principles



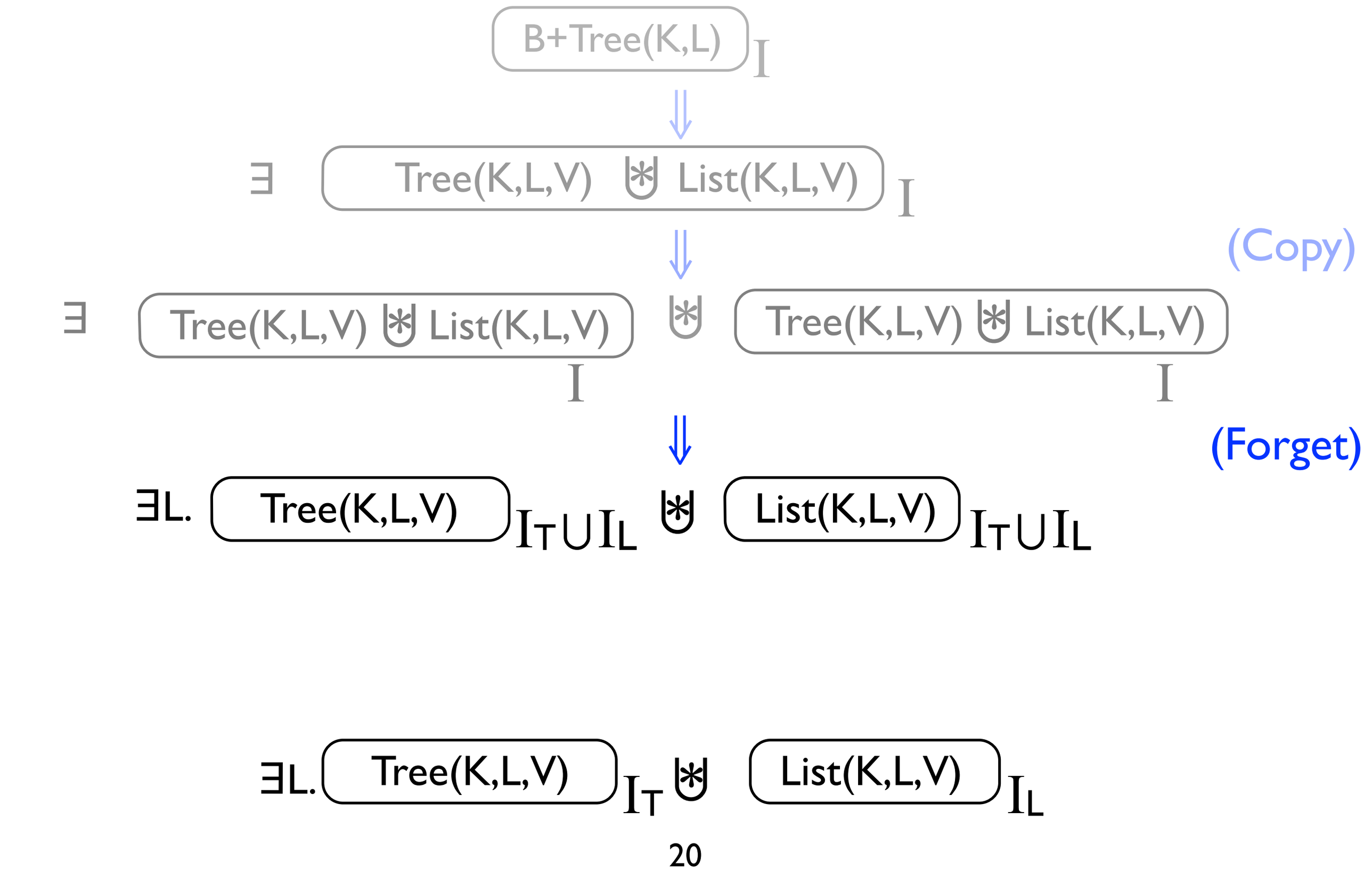
Forgetting Resources



CoLoSL Principles



CoLoSL Principles

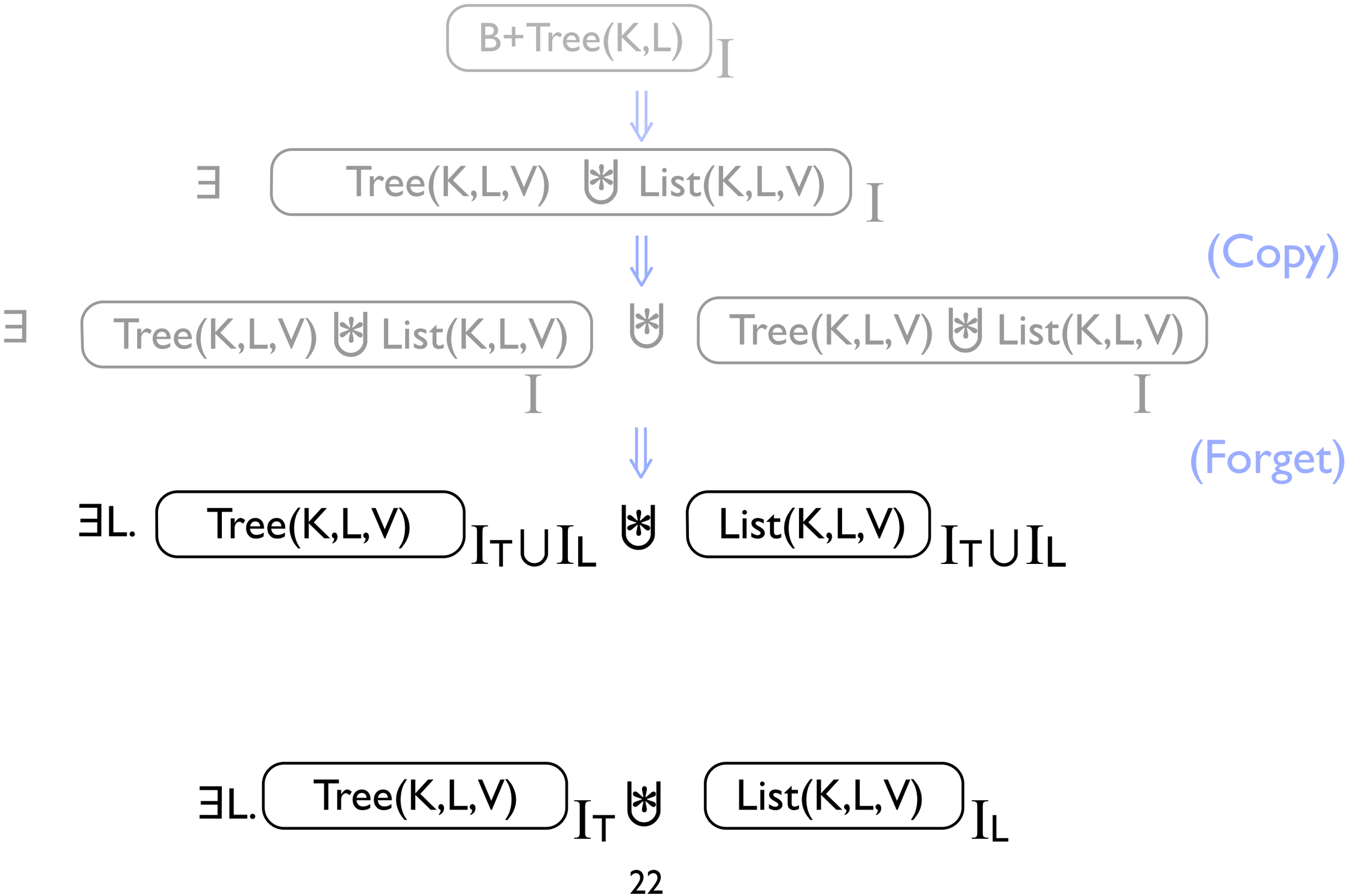


Forgetting Interference (Shift)

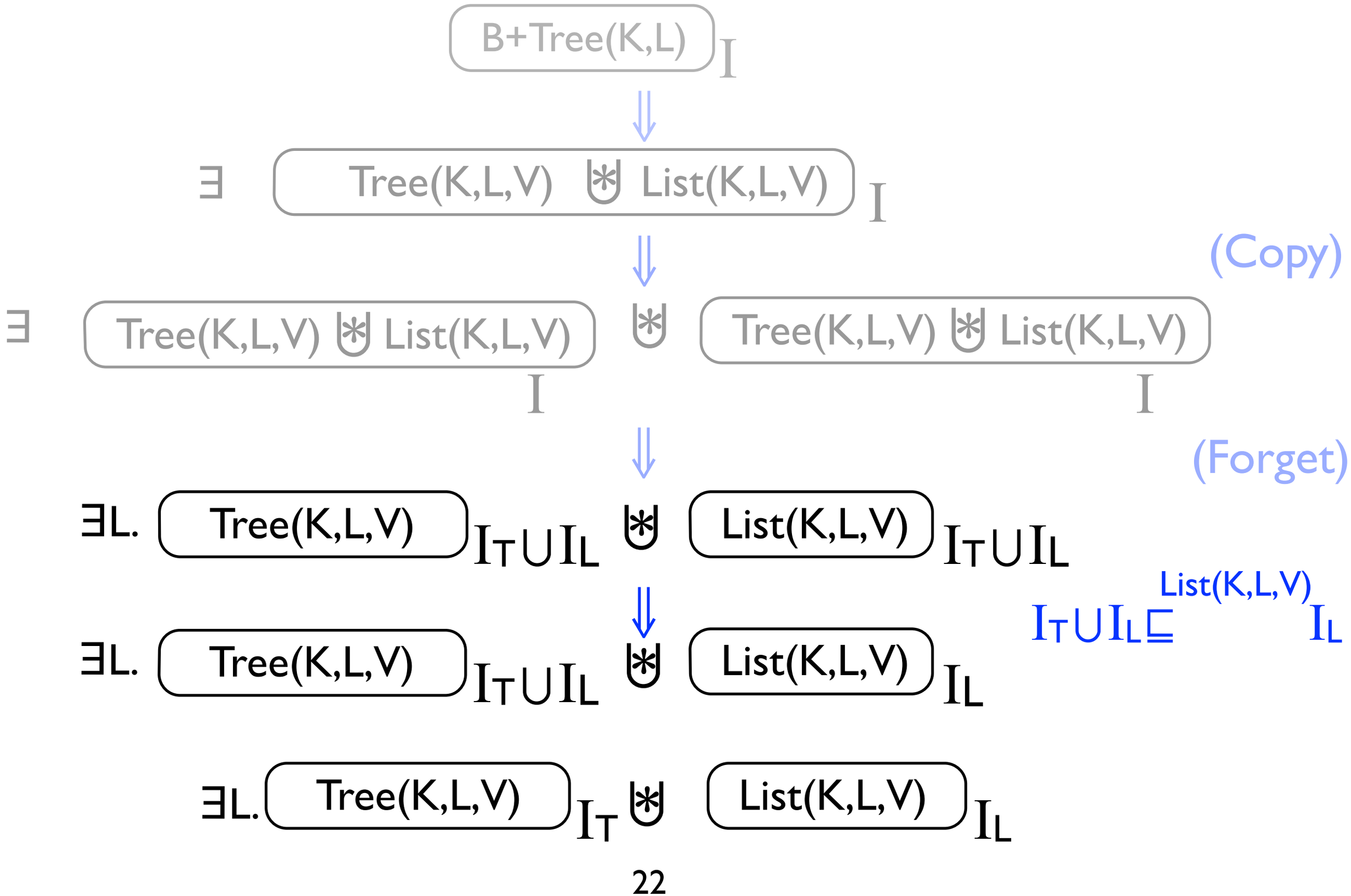
if $I \sqsubseteq^{P} I'$

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

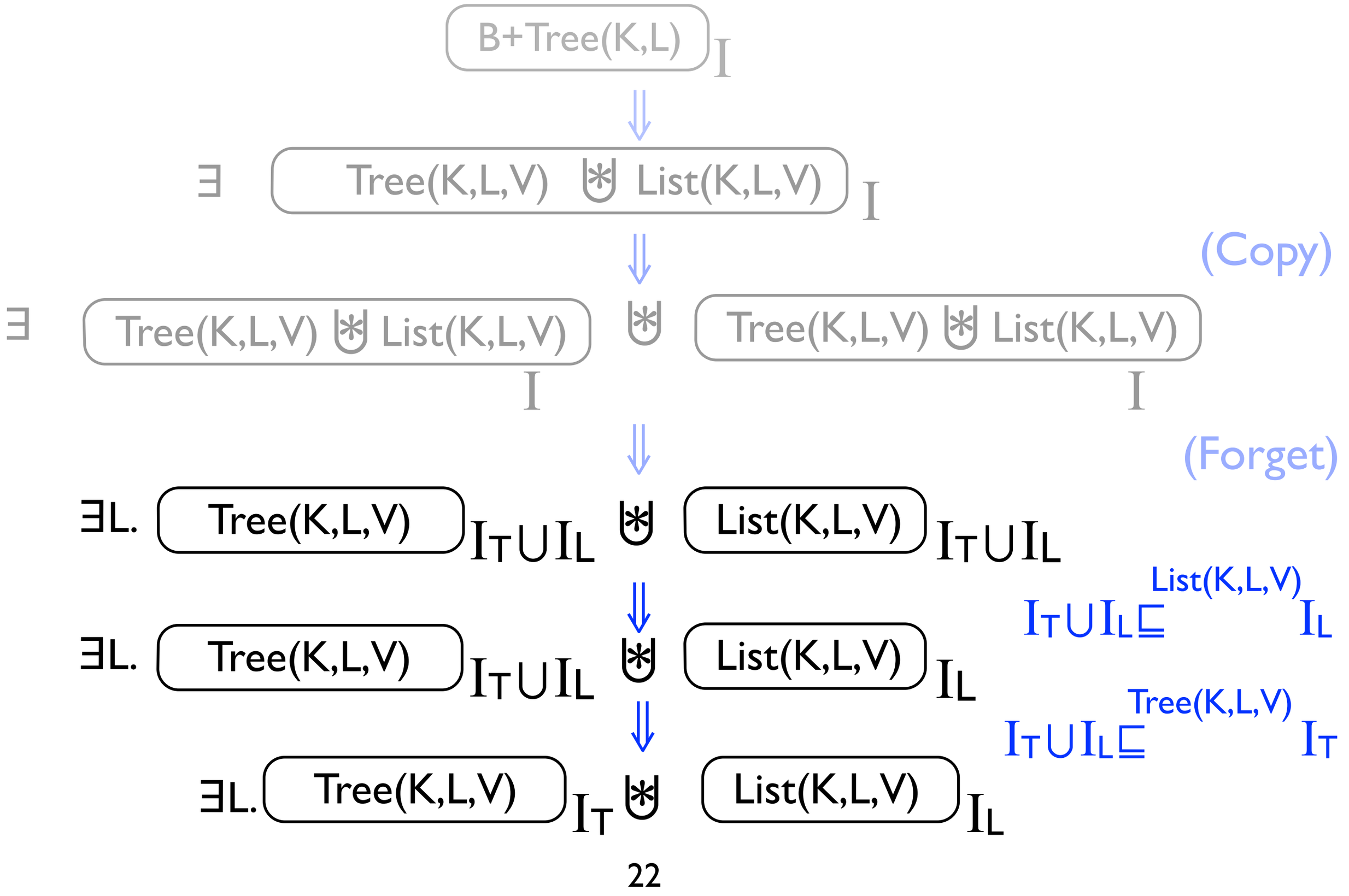
CoLoSL Principles



CoLoSL Principles

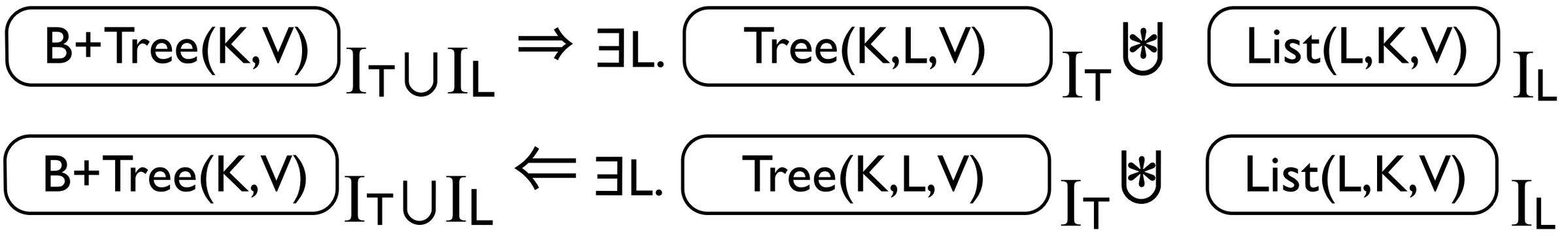


CoLoSL Principles

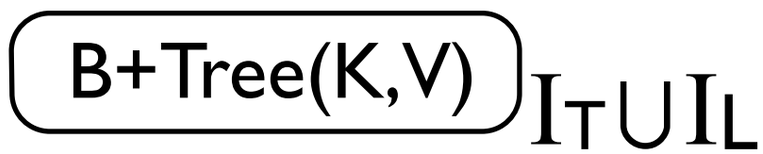


CoLoSL Principles

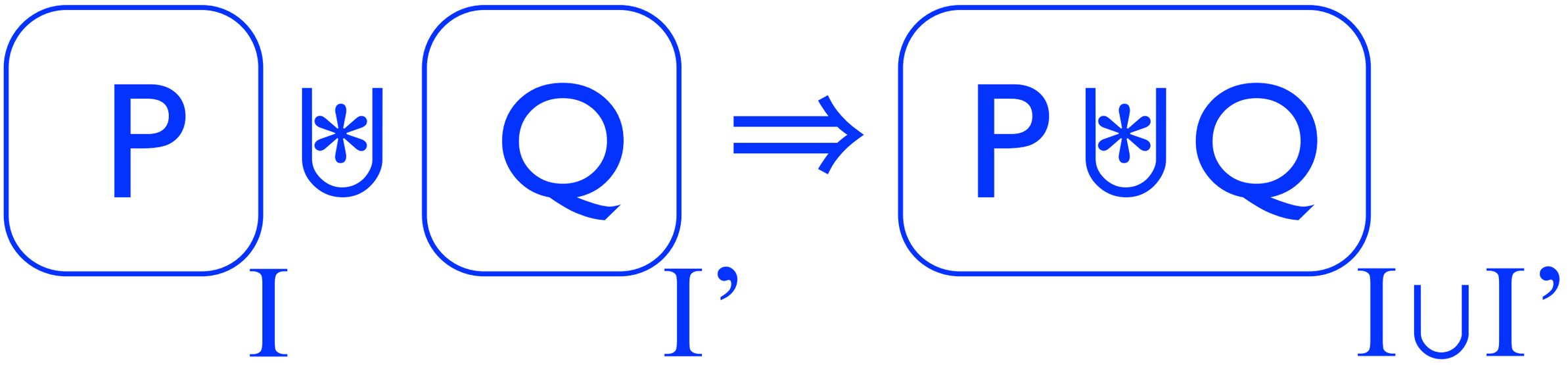
CoLoSL Principles



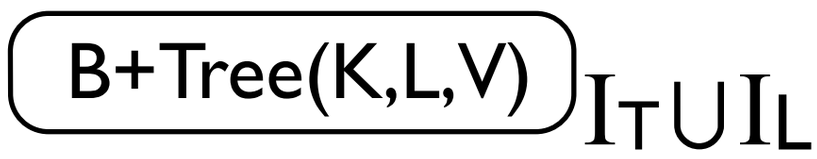
CoLoSL Principles



Merging Resources



CoLoSL Principles



CoLoSL Principles

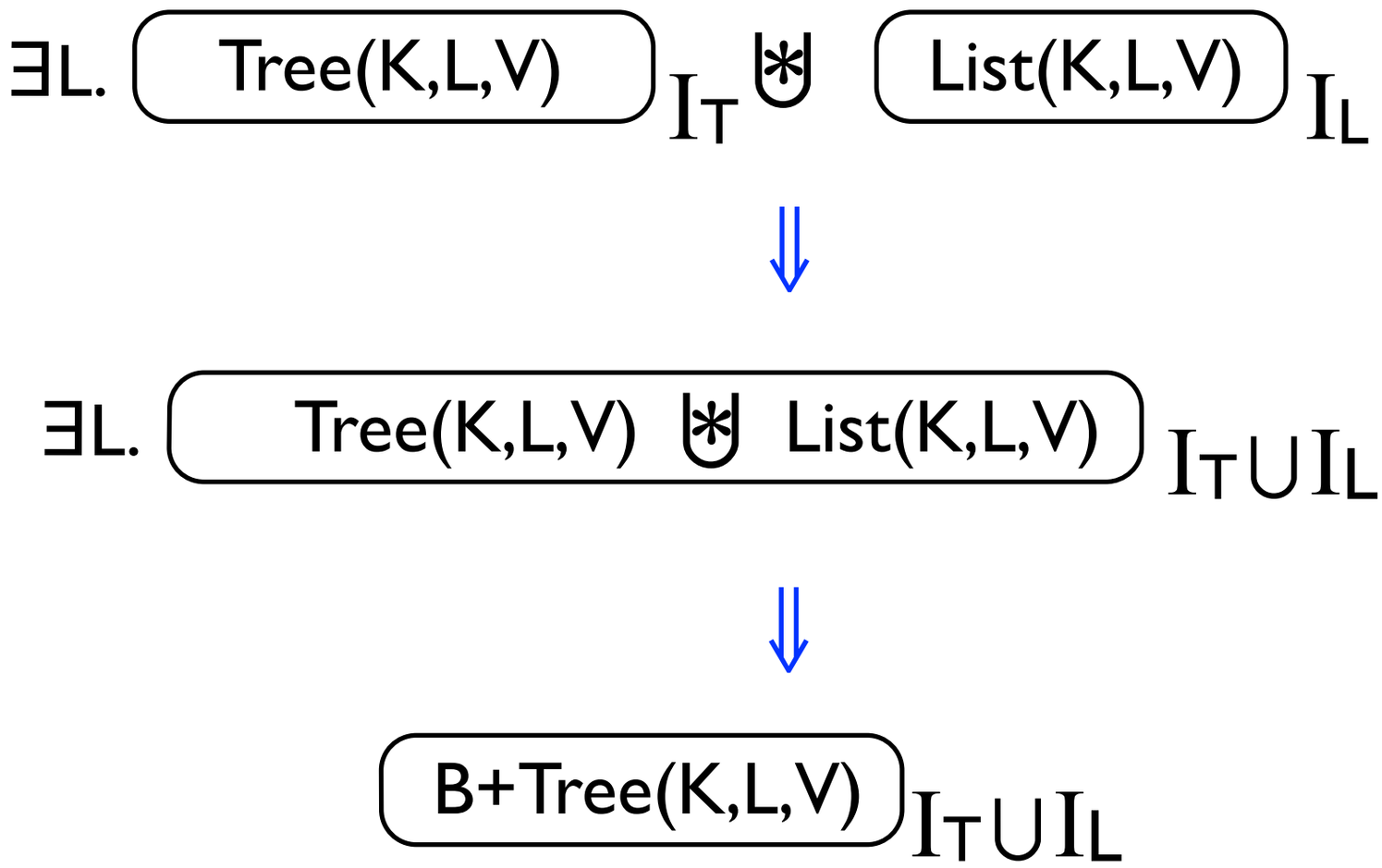
$\exists L.$ Tree(K,L,V) $_{I_T}$ ⍟ List(K,L,V) $_{I_L}$

⇓ (Merge)

$\exists L.$ Tree(K,L,V) ⍟ List(K,L,V) $_{I_T \cup I_L}$

B+Tree(K,L,V) $_{I_T \cup I_L}$

CoLoSL Principles



Examples

- B+ Tree

- **Concurrent List**
  - Dynamic extension

Examples

- B+ Tree
- Concurrent List
  - Dynamic extension
- **Spanning Tree**
  - Recursive overlapping graph predicate
  - Local proof; proof structures matches the algorithm

Examples

- B+ Tree
- Concurrent List
  - Dynamic extension
- Spanning Tree
  - Recursive overlapping graph predicate
  - Local proof; proof structures matches the algorithm
- **Dijkstra's Self-stabilising Token Ring**
  - Local proof; proof reuse

Conclusions and Future Work

- CoLoSL
  - Subjective/overlapping views
  - Interference composition ☞ more flexible framing
  - Dynamic extension
  - Are we there yet? No!

Conclusions and Future Work

- CoLoSL
  - Subjective/overlapping views
  - Interference composition ☞ more flexible framing
  - Dynamic extension
  - Are we there yet? No!
- Future Work
  - Abstract predicates, abstract atomicity, …
  - CoLoSL in Iris (monoid/invariant to split the interference)

Thank you for listening!