A GrandSib-Crossing Parser Invariants

Each type of sub-problem is associated with both *interior* and *exterior* invariants (involving specified edges, bad (or not bad) contexts, or crossings). When two or more sub-problems are combined, their interior and exterior invariants are also combined to produce new interior and exterior invariants for the larger combined subforest. Table 7 shows the full set of interior and exterior invariants for each type of sub-problem. For ease of exposition, left versions are given for asymmetric cases (i.e., $dir = L$); the right versions are symmetric.

Each crossing region of Pitler et al. (2013) has a signature of $\text{type}[i, j, x, b_i, b_j, b_x]$. i, j, and x are in $[0, n]$ and indicate that the sub-problem is over vertices $[i, j] \cup \{x\}$; i and/or j can be the crossing point for edges between x and (i, j); b_i, b_j, and b_x are booleans indicating whether i, j, or x, respectively, receives its incoming edge from its parent within this sub-problem. This signature is used to divide the interval vertices $[i, j]$ into those that can be directly connected by an edge to x in this sub-problem (and such an edge is guaranteed to be crossed) and those that cannot be directly connected to x.

Definition 4. *$VCross(i, j, b_i, b_j, type)$ is the interval of vertices that are neither roots nor crossing points of the sub-problem. All internal vertices in (i, j) are never roots nor crossing points, yielding four possible cases that vary in whether the boundary vertices i and/or j are included. In particular:*

$$VCross(i, j, b_i, b_j, type) = \begin{cases} (i, j) & : type = LR \vee \left(\begin{array}{l} (type = L \vee b_i = F) \\ \wedge (type = R \vee b_j = F)) \end{array} \right) \\ [i, j) & : b_i = T \wedge (type = R \vee (type = N \wedge b_j = F)) \\ (i, j] & : b_j = T \wedge (type = L \vee (type = N \wedge b_i = F)) \\ [i, j] & : type = N \wedge b_i = T \wedge b_j = T \end{cases}$$

The Crossing Conditions (Definition 5) are enforced by construction, and taken together ensure that whenever a crossing region is used, edges between the exterior point and the interval are crossed.

Definition 5. *The* Crossing Conditions *of a crossing region $type[i, j, x, b_i, b_j, b_x]$ are:*

- *All edges in the sub-problem incident to x have the other endpoint of the edge in $VCross(i, j, b_i, b_j, type)$*
- *There exists at least one edge between x and $VCross(i, j, b_i, b_j, type)$*
- *All edges between x and vertices in $VCross(i, j, b_i, b_j, type)$ will be crossed by an edge outside of the sub-problem.*

	Interior	Exterior
TriG[i,j,g,L]	• Rooted at: i	• Includes edge $\vec{e}_{gi}$ • $\neg BadContext_L(i, j)$
Tri[i,j,L]	• Rooted at: i	• $BadContext_L(i, j)$
TrapG[i,j,g,L]	• Rooted at: i • Includes edge $\vec{e}_{ij}$	• Includes edge $\vec{e}_{gi}$ • $\neg BadContext_L(i, j)$
BoxG[i,j,g]	• Rooted at: i,j	• Includes edge $\vec{e}_{gi}$; $\vec{e}_{gi}$ not crossed • Includes edge $\vec{e}_{gj}$; $\vec{e}_{gj}$ not crossed • i and j are adjacent children on the same side of g
Trap[i,j,L]	• Rooted at: i • Includes edge $\vec{e}_{ij}$	• $BadContext_L(i, j)$

TwoRooted[i,j]	• Rooted at: i, j	• $BadContext_L(i, j)$ • $BadContext_R(i, j)$
OneFarCrossedG[i,j,g,L]	• Rooted at: i, j • Edge from j to leftmost child in (i, j) is crossed	• Includes edge $\vec{e}_{gi}$ • $\neg BadContext_L(i, j)$
LeftFarCrossed[i,j,L]	• Rooted at: i, j • Edge from i to rightmost child in (i, j) is crossed	• $BadContext_R(j, i)$
Chain[i,j]	• Rooted at: i, j • Edge from i to i's furthest child and edge from j to j's furthest child part of same chain of crossing edges	
TriFarCrossed[i,j,L]	• Rooted at: i • Edge from i to rightmost child in (i, j) crossed	
Page1[i,j,x]	• Rooted at: j, x • i, j, x involved in same chain of crossing edges • x has at least one child in (i, j) • i's parent is to the right of x's child or children	
Page2[i,j,x]	• Rooted at: i, j • i, j, x involved in same chain of crossing edges • x's parent in (i, j) • x has no children in $[i, j]$ • i has at least one child to the right of x's parent	
Chain_JFromI[i,j,x]	• Rooted at: i, x • i, j, x involved in same chain of crossing edges • j is descended from i • If i is not the parent of j, then x has exactly one child in (i, j)	
Chain_JFromX[i,j,x]	• Rooted at: i, x • i, j, x involved in same chain of crossing edges • j is descended from x • x has exactly one child in (i, j)	

LR[i,j,x,b_x]	• Rooted at: i, j, $1_{[b_x=F]}x$ • Crossing Conditions • Exists some split point such that between x and left of the split point are only crossed by edges incident to i, and to the right only those incident to j	• Includes edge $\vec{e}_{ij}$ or edge $\vec{e}_{ji}$ • No other edge besides $\vec{e}_{ij}$/$\vec{e}_{ji}$ will cross the edges between x and $VCross(i, j, F, F, LR)$
L[i,j,x,b_i,b_j,b_x]	• Rooted at: $1(b_i = F)i$, $1(b_j = F)j$, $1(b_x = F)x$ • Crossing Conditions • Edges between x and $VCross(i, j, b_i, b_j, L)$ have i as their crossing point	• Exists some external edge incident to i that will cross all edges between x and $VCross(i, j, b_i, b_j, L)$ • If $b_i = F$, $BadContext_L(i, j)$ • If $b_j = F$, $BadContext_R(i, j)$
N[i,j,x,b_i,b_j,b_x]	• Rooted at: $1(b_i = F)i$, $1(b_j = F)j$, $1(b_x = F)x$ • Crossing Conditions • No edges between x and $VCross(i, j, b_i, b_j, N)$ are crossed	• Exists at least two external edges that force a point outside of $\{i, j\}$ to be the crossing point for the edges between x and $VCross(i, j, b_i, b_j, N)$ • If $b_i = F$, $BadContext_L(i, j)$ • If $b_j = F$, $BadContext_R(i, j)$

Table 7: Internal and External Invariants

B Full Dynamic Program for the GrandSib-Crossing Parser

The initial sub-problems are:
$\forall i Tri[i,i,L] = Tri[i,i,R] = TwoRooted[i,i] = TwoRooted[i,i+1] = 0$,
$\forall g,h > g \quad TriG[h,h,g,L] = GCO1(g,h,NIL) \quad TriG[h,h,g,R] = GCI1(g,h,NIL)$,
$\forall g,h < g \quad TriG[h,h,g,L] = GCI1(g,h,NIL) \quad TriG[h,h,g,R] = GCO(g,h,NIL)$.[4] The final tree is in: $TriG[0,n,-1,L]$ (where 0 is the artificial root and -1 is an artificial parent of the root). All sub-problems left undefined (for example, $Int[i,j,T,T]$, in which all vertices must get their parent from the interval, or $L[i,i,x,b_i,b_j,b_x]$, which would have $VCross(i,i,b_i,b_j,L) = \emptyset$) have a score of $-\infty$.

$\text{Score}(\text{GrandSib}(\text{g},\text{h},\text{m},\text{s}))$ is abbreviated as $GS(g,h,m,s)$, $\text{Score}(\text{Grand}(\text{g},\text{h},\text{m}))$ as $G(g,h,m)$, $\text{Score}(\text{Sib}(\text{h},\text{m},\text{s}))$ as $S(h,m,s)$, $\text{Score}(\text{Edge}(\text{h},\text{m}))$ as $E(h,m)$, and $\text{Score}(\text{CrossedEdge}(\text{h},\text{m}))$ as $CE(h,m)$.

The implemented parser distinguishes between inner and outer grandchildren; this can be determined from the indices of g, h, and m above.

Following the notation of Pitler et al. (2013), $TF(b,S)$ means that if $b = T$, exactly one of the booleans in the set S must be true; if $b = F$, none of the booleans in S can be true. For example, $TF(b_i,\{b_l,b_r\})$ would mean: if $b_i = T$, either ($b_l = T$ and $b_r = F$) or ($b_l = F$ and $b_r = T$), while if $b_i = F$, then $b_l = b_r = F$. These are used for setting the booleans corresponding to whether a node has an incoming edge (parent) within a particular sub-problem.

Note that "triangles" have the same semantics as "intervals" where only one boundary point needs a parent. To emphasize the similarities to both the dynamic programs of Koo and Collins (2010) and Pitler et al. (2013), the most familiar vocabulary in each section is used, as well as the following syntactic sugar:
$TwoRooted[i,j] := Int[i,j,F,F]$, $Tri[i,j,L] := Int[i,j,F,T]$, and
$Tri[i,j,R] := Int[i,j,T,F]$.

[4] These correspond to the null boundary cases described in Koo (2010, Appendix B.1.5).
These are used when $\vec{e}_{gh}$ is not crossed and there are no $GProj$ edges from h to any interior children (and so the score $GCI(g,h,NIL)$ is added); similarly when $\vec{e}_{gh}$ is not crossed and there are no $GProj$ edges from h to any exterior children ($GCO(g,h,NIL)$). In the projective case, since $\vec{e}_{gh}$ is never crossed and all edges to children are $GProj$, this simplifies to h having *no* inner (outer) children. In a non-projective tree, this covers both the case in which there are no children and the case in which the edge to the most outer interior/exterior child is crossed.

$TriG[i,j,g,L] \leftarrow \max$
$$\begin{cases} \text{// Furthest child of } i \text{ not crossed} \\ \max_{k\in(i,j]} TrapG[i,k,g,L] + TriG[k,j,i,L] \\ \\ \text{// Furthest child of } i \text{ crossed} \\ TriFarCrossed[i,j,L] \end{cases}$$

$TriG[i,j,g,R] \leftarrow$ symmetric to above

$Tri[i,j,L] \leftarrow \max$
$$\begin{cases} \text{// Furthest child of } i \text{ not crossed} \\ \max_{k\in(i,j]} Trap[i,k,L] + TriG[k,j,i,L] \\ \\ \text{// Furthest child of } i \text{ crossed} \\ TriFarCrossed[i,j,L] \end{cases}$$

$Tri[i,j,R] \leftarrow$ symmetric to above

$TrapG[i,j,g,L] \leftarrow \max$
$$\begin{cases} \text{// } j \text{ is the innermost child of } i \\ GS(g,i,j,-) + TriG[i+1,j,i,R] \\ \\ \text{// The chain of crossing edges including } j\text{'s inner sibling involves edges from } j \\ G(g,i,j) + Chain[i,j] \\ \\ \text{// } k \text{ is } j\text{'s inner sibling and } \vec{e}_{ik} \text{ is not crossed} \\ \max_{k\in(i,j)} GS(g,i,j,k) + TrapG[i,k,g,L] + BoxG[k,j,i] \\ \\ \text{// The edge to } j\text{'s inner sibling is crossed but the chain does not involve any descendants of } j \\ \max_{k\in(i,j)} G(g,i,j) + TriFarCrossed[i,k,L] + TriG[k+1,j,i,R] \\ \\ \text{// The edge to } j\text{'s inner sibling is crossed and the chain involves descendants of } j \\ \max_{k\in(i,j)} G(g,i,j) + Chain[i,k] + TriG[k,j,i,R] \end{cases}$$

$TrapG[i,j,g,R] \leftarrow$ symmetric to above

$Trap[i, j, L] \leftarrow \max$
$\begin{cases} \text{// } j \text{ is the innermost child of } i \\ S(i, j, -) + TriG[i + 1, j, i, R] \\ \\ \text{// The chain of crossing edges including } j\text{'s inner sibling involves edges from } j \\ E(i, j) + Chain[i, j] \\ \\ \text{// } k \text{ is } j\text{'s inner sibling and } \vec{e}_{ik} \text{ is not crossed} \\ \max_{k \in (i,j)} S(i, j, k) + Trap[i, k, L] + BoxG[k, j, i] \\ \\ \text{// The edge to } j\text{'s inner sibling is crossed but the chain does not involve any descendants of } j \\ \max_{k \in (i,j)} E(i, j) + TriFarCrossed[i, k, L] + TriG[k + 1, j, i, R] \\ \\ \text{// The edge to } j\text{'s inner sibling is crossed and the chain involves descendants of } j \\ \max_{k \in (i,j)} E(i, j) + Chain[i, k] + TriG[k, j, i, R] \end{cases}$

$Trap[i, j, R] \leftarrow$ symmetric to above

$BoxG[i, j, g] \leftarrow \max$
$\begin{cases} \text{// The descendants of } i \text{ and } j \text{ do not interleave} \\ \max_{k \in [i,j)} TriG[i, k, g, L] + TriG[k + 1, j, g, R] \\ \\ \text{// The descendants of } i \text{ and } j \text{ interleave} \\ \max_{k \in [i,j)} TriG[i, k, g, L] + OneFarCrossedG[k, j, g, R] \end{cases}$

// k is a descendant of i involved in the chain of crossing edges with j
$OneFarCrossedG[i, j, g, L] \leftarrow \max_{k \in [i,j)} TriG[i, k, g, L] + Chain[k, j]$

$OneFarCrossedG[i, j, g, R] \leftarrow$ symmetric to above

// k is a descendant of j involved in the chain of crossing edges with i
$LeftFarCrossed[i, j] \leftarrow \max_{k \in (i,j]} Chain[i, k] + Tri[k, j, R]$

$TwoRooted[i, j] \leftarrow \max$
$\begin{cases} \text{// The descendants of } i \text{ and } j \text{ do not interleave} \\ \max_{k \in [i,j)} Tri[i, k, L] + Tri[k + 1, j, R] \\ \\ \text{// The edge from } k \text{ to } k\text{'s furthest child is crossed, } j\text{'s descendants involved in the crossing chain} \\ \max_{k \in [i,j)} Tri[i, k, L] + LeftFarCrossed[k, j] \end{cases}$

$Chain[i,j] \leftarrow \max$

- // Edges incident to i and j cross directly, j is the crossing point for $\vec{e}_{ik}$
  $\max_{k \in (i,j)} CE(i,k) + LR[i,k,j,F] + TwoRooted[k,j]$
- // Edges incident to i and j cross directly, l is the crossing point for $\vec{e}_{ik}$
  // and i has children in (l,k)
  $\max_{k \in (i,j),\ l \in (i,k)} CE(i,k) + CE(j,l) + TwoRooted[i,l] + L[l,k,i,F,F,F] + N[k,j,l,F,F,F]$
- // and i has no children in (l,k), k has no children in (i,l)
  $\max_{k \in (i,j),\ l \in (i,k)} CE(i,k) + CE(j,l) + TwoRooted[i,l] + TwoRooted[l,k] + L[k,j,l,F,F,F]$
- // and i has no children in (l,k), k has children in (i,l)
  $\max_{k \in (i,j),\ l \in (i,k)} CE(i,k) + CE(j,l) + R[i,l,k,F,F,F] + TwoRooted[l,k] + L[k,j,l,F,F,F]$
- // Edges incident to i and j do not cross directly
  $\max_{k \in (i,j),\ l \in (k,j)} CE(i,k) + R[i,k,l,F,F,F] + TwoRooted[k,l] + Page1[l,j,k]$

// Rooted at x and j, the edge from i's parent to i crosses an edge from x to its child
$Page1[i,j,x] \leftarrow \max$

- // End of the chain $\rightarrow j$ is i's parent
  $CE(j,i) + L[i,j,x,F,F,F]$
- // Chain continuing
  $\max_{k \in (i,j)} CE(x,k) + TwoRooted[i,k] + Page2[k,j,i]$

// Rooted at i and j, the edge from i to a child crosses an edge from x's parent to x
$Page2[i,j,x] \leftarrow \max_{k \in (i,j)} CE(k,x) + TwoRooted[i,k] + Page1[k,j,i]$

$TriFarCrossed[i,j,L] \leftarrow \max$

- // k has edges into $(l,j]$
  $\max_{k \in (i,j),\ l \in (k,j],\ TF(T,\{b_l,b_m,b_r\})} CE(i,k) + R[i,k,l,F,F,b_l] + Int[k,l,F,b_m] + L[l,j,k,b_r,T,F]$
- // k does not have edges into $(l,j]$
  $\max_{k \in (i,j),\ l \in (k,j],\ TF(T,\{b_l,b_m,b_r\})} CE(i,k) + LR[i,k,l,b_l] + Int[k,l,F,b_m] + Int[l,j,b_r,T]$
- // i has edges into (l,k)
  $\max_{k \in (i,j),\ l \in (i,k),\ TF(T,\{b_l,b_m,b_r\})} CE(i,k) + Int[i,l,F,b_l] + L[l,k,i,b_m,F,F] + N[k,j,l,F,T,b_r]$
- // i does not have edges into (l,k), k does not have edges into (i,l)
  $\max_{k \in (i,j),\ l \in (i,k),\ TF(T,\{b_l,b_m,b_r\})} CE(i,k) + Int[i,l,F,b_l] + Int[l,k,b_m,F] + L[k,j,l,F,T,b_r]$
- // i does not have edges into (l,k), k has edges into (i,l)
  $\max_{k \in (i,j),\ l \in (i,k),\ TF(T,\{b_l,b_m,b_r\})} CE(i,k) + R[i,l,k,F,b_l,F] + Int[l,k,b_m,F] + L[k,j,l,F,T,b_r]$

$TriFarCrossed[i,j,R] \leftarrow$ symmetric to above

$LR[i, j, x, b_x] \leftarrow \max$

- // No edges incident to j cross edges from x
  $L[i, j, x, F, F, b_x]$
- // No edges incident to i cross edges from x
  $R[i, j, x, F, F, b_x]$
- // x does not get its parent from $[i, k]$, edge from k's parent to k is crossed
  // and k is descended from i
  $\max_{k \in (i,j)} Chain\_JFromI[i, k, x] + R[k, j, x, F, F, b_x]$
- // and k is descended from x, x is descended from j
  $\max_{k \in (i,j)} Chain\_JFromX[i, k, x] + R\_XFromJ[k, j, x]$ if $b_x = T$
- // and k is descended from x, x's parent is elsewhere
  $\max_{k \in (i,j)} Chain\_JFromX[i, k, x] + R[k, j, x, F, F, F]$ if $b_x = F$
- // x does not get its parent from $[k, j]$, edge from k's parent to k is crossed
  // and k is descended from j
  $\max_{k \in (i,j)} L[i, k, x, F, F, b_x] + Chain\_IFromJ[k, j, x]$
- // and k is descended from x, x is descended from i
  $\max_{k \in (i,j)} L\_XFromI[i, k, x] + Chain\_IFromX[k, j, x]$ if $b_x = T$
- // and k is descended from x, x's parent is elsewhere
  $\max_{k \in (i,j)} L[i, k, x, F, F, F] + Chain\_IFromX[k, j, x]$ if $b_x = F$

// Chain of edges from i through j, rooted at i and x
$Chain\_JFromI[i, j, x] \leftarrow \max$

- // End of the chain $\rightarrow$ i is j's parent
  $CE(i, j) + L[i, j, x, F, F, F]$
- // Chain continuing
  $\max_{k \in (i,j)} CE(x, k) + Int[i, k, F, F] + Chain\_JFromX[k, j, i, L]$

$Chain\_IFromJ[i, j, x] \leftarrow$ symmetric to above

// Chain of edges from i through j, rooted at i and x
$Chain\_JFromX[i, j, x] \leftarrow \max_{k \in (i,j)} CE(x, k) + Int[i, k, F, F] + Chain\_JFromI[k, j, i]$

$Chain\_IFromX[i, j, x] \leftarrow$ symmetric to above

$N[i, j, x, b_i, b_j, F] \leftarrow \max$

$\begin{cases} \text{// } x \text{ has no more edges into interior} \\ \max_{k \in VCross(i,j,b_i,b_j,N)} CE(x,k) + Int[i,k,b_i,F] + Int[k,j,F,b_j] \\ \\ \text{// } x \text{ has at least one more edge into interior} \\ \max_{k \in VCross(i,j,b_i,b_j,N)} CE(x,k) + N[i,k,x,b_i,F,F] + Int[k,j,F,b_j] \end{cases}$

$N[i, j, x, b_i, b_j, T] \leftarrow \max$

$\begin{cases} \text{// No more edges with } x \text{ on either side} \\ \max_{k \in VCross(i,j,b_i,b_j,N),\ TF(T,\{b_l,b_r\})} CE(k,x) + Int[i,k,b_i,b_l] + Int[k,j,b_r,b_j] \\ \\ \text{// } x \text{ has more children on the left} \rightarrow k\text{'s parent is on the right and } x \text{ has no more children on the right} \\ \max_{k \in VCross(i,j,b_i,b_j,N)} CE(k,x) + N[i,k,x,b_i,F,F] + Int[k,j,T,b_j] \\ \\ \text{// } x \text{ has more children on the right} \rightarrow k\text{'s parent is on the left and } x \text{ has no more children on the left} \\ \max_{k \in VCross(i,j,b_i,b_j,N)} CE(k,x) + Int[i,k,b_i,T] + N[k,j,x,F,b_j,F] \end{cases}$

$L[i, j, x, b_i, b_j, F] \leftarrow \max$

$\begin{cases} \text{// Only one edge with } x \text{ and no edges from } i \text{ into } (k,j] \\ \max_{k \in VCross(i,j,b_i,b_j,L)} CE(x,k) + Int[i,k,b_i,F] + Int[k,j,F,b_j] \\ \\ \text{// Only one edge with } x \text{ and there are edges from } i \text{ into } (k,j] \\ \max_{k \in VCross(i,j,b_i,b_j,L),\ TF(b_i,\{b_l,b_r\})} CE(x,k) + Int[i,k,b_l,F] + L[k,j,i,F,b_j,b_r] \\ \\ \text{// Multiple edges with } x\text{: Choose furthest edge incident to } i \text{ or } x \\ \text{// More edges with } x \text{ and no edges from } i \text{ into } (k,j] \\ \max_{k \in VCross(i,j,b_i,b_j,L)} CE(x,k) + L[i,k,x,b_i,F,F] + Int[k,j,F,b_j] \\ \\ \text{// No edges from } x \text{ into } (k,j] \\ \max_{k \in VCross(i,j,b_i,b_j,L)} CE(i,k) + L[i,k,x,F,F,F] + Int[k,j,F,b_j] \qquad \text{if } b_i = F \\ \\ \text{// No edges from } x \text{ into } (k,j] \\ \max_{k \in VCross(i,j,b_i,b_j,L)} CE(i,k) + L\_IFromX[i,k,x] + Int[k,j,F,b_j] \qquad \text{if } b_i = T \\ \\ \text{// No edges from } x \text{ into } (k,j],\ k\text{'s parent is in } (i,k) \cup \{x\} \\ \max_{k \in VCross(i,j,b_i,b_j,L)} CE(k,i) + L\_JFromX[i,k,x] + Int[k,j,F,b_j] \qquad \text{if } b_i = T \\ \\ \text{// No edges from } x \text{ into } (k,j],\ k\text{'s parent is in } (k,j] \\ \max_{k \in VCross(i,j,b_i,b_j,L)} CE(k,i) + L[i,k,x,F,F,F] + Int[k,j,T,b_j] \qquad \text{if } b_i = T \end{cases}$

$L[i,j,x,F,b_j,T] \leftarrow \max$

// Only one edges with $x \rightarrow$ edge from x's parent
// No edges from i into $(k,j]$
$\max_{k \in VCross(i,j,F,b_j,L),\ TF(T,\{b_l,b_r\})} CE(k,x) + Int[i,k,F,b_l] + Int[k,j,b_r,b_j]$

// Edges from i into $(k,j]$
$\max_{k \in VCross(i,j,F,b_j,L),\ TF(T,\{b_l,b_r\})} CE(k,x) + Int[i,k,F,b_l] + L[k,j,i,b_r,b_j,F]$

// Multiple edges with x: Choose furthest edge incident to i or x
// Furthest edge is the parent of x and k's parent is from the left
$\max_{k \in VCross(i,j,F,b_j,L)} CE(k,x) + L\_JFromI[i,k,x] + Int[k,j,F,b_j]$

// Furthest edge is the parent of x and k's parent is from the right
$\max_{k \in VCross(i,j,F,b_j,L)} CE(k,x) + L[i,k,x,F,F,F] + Int[k,j,T,b_j]$

// Furthest edge is a child of x
$\max_{k \in VCross(i,j,F,b_j,L)} CE(x,k) + L\_XFromI[i,k,x] + Int[k,j,F,b_j]$

// Furthest edge is a child of i
$\max_{k \in VCross(i,j,F,b_j,L)} CE(i,k) + L[i,k,x,F,F,T] + Int[k,j,F,b_j]$

$L\_XFromI[i,j,x] \leftarrow \max$

// Only one edge with $x \rightarrow$ edge from x's parent to x
// No edges from i into (k,j)
$\max_{k \in (i,j)} CE(k,x) + Int[i,k,F,T] + Int[k,j,F,F]$

// Edges from i into (k,j), k's parent from its left
$\max_{k \in (i,j)} CE(k,x) + Int[i,k,F,T] + L[k,j,i,F,F,F]$

// Edges from i into (k,j), k's parent from its right
$\max_{k \in (i,j)} CE(k,x) + Int[i,k,F,F] + L\_IFromX[k,j,i]$

// Multiple edges with x: Choose furthest edge incident to i or x
// Furthest edge is the parent of x
$\max_{k \in (i,j)} CE(k,x) + L\_JFromI[i,k,x] + Int[k,j,F,F]$

// Furthest edge is a child of x
$\max_{k \in (i,j)} CE(x,k) + L\_XFromI[i,k,x] + Int[k,j,F,F]$

// Furthest edge is a child of i
$\max_{k \in (i,j)} CE(i,k) + L[i,k,x,F,F,T] + Int[k,j,F,F]$

$L\_IFromX[i,j,x] \leftarrow \max$

- // Only one edge with x, no edges from i into (k,j)
  $\max_{k\in(i,j)} CE(x,k) + Int[i,k,T,F] + Int[k,j,F,F]$
- // Only one edge with x, edges from i into (k,j), i's parent in $(i,k]$
  $\max_{k\in(i,j)} CE(x,k) + Int[i,k,T,F] + L[k,j,i,F,F,F]$
- // Only one edge with x, edges from i into (k,j), i's parent in (k,j)
  $\max_{k\in(i,j)} CE(x,k) + Int[i,k,F,F] + L\_XFromI[k,j,i]$
- // Multiple edges with x: Choose furthest edge incident to i or x
  // Furthest edge is to x's child
  $\max_{k\in(i,j)} CE(x,k) + L[i,k,x,T,F,F] + Int[k,j,F,F]$
- // Furthest edge is from i's parent
  $\max_{k\in(i,j)} CE(k,i) + L\_JFromX[i,k,x] + Int[k,j,F,F]$
- // Furthest edge is to i's child
  $\max_{k\in(i,j)} CE(i,k) + L\_IFromX[i,k,x] + Int[k,j,F,F]$

$L\_JFromX[i,j,x] \leftarrow \max$

- // Only one edge with x and no edges from i into (k,j)
  $\max_{k\in(i,j]} CE(x,k) + Int[i,k,F,F] + Int[k,j,F,T]$
- // Only one edge with x and edges from i into (k,j)
  $\max_{k\in(i,j]} CE(x,k) + Int[i,k,F,F] + L\_JFromI[k,j,i]$
- // More edges with $x \rightarrow$ furthest edge is to x's child
  $\max_{k\in(i,j]} CE(x,k) + L[i,k,x,F,F,F] + Int[k,j,F,T]$

$L\_JFromI[i,j,x] \leftarrow \max$

- // Only one child of $x \rightarrow$ edges from i into (k,j)
  $\max_{k\in(i,j)} CE(x,k) + Int[i,k,F,F] + L\_JFromX[k,j,i]$
- // Multiple children of $x \rightarrow$ furthest edge incident to x or i is an edge from i
  $\max_{k\in(i,j]} CE(i,k) + L[i,k,x,F,F,F] + Int[k,j,F,T]$

$R[i,j,x,b_i,b_j,F] \leftarrow$ symmetric to $L[i,j,x,b_i,b_j,F]$

$R[i,j,x,b_i,F,T] \leftarrow$ symmetric to $L[i,j,x,F,b_j,T]$

$R\_XFromJ[i,j,x] \leftarrow$ symmetric to $L\_XFromI[i,j,x]$

$R\_JFromX[i,j,x] \leftarrow$ symmetric to $L\_IFromX[i,j,x]$

$R\_IFromX[i,j,x] \leftarrow$ symmetric to $L\_JFromX[i,j,x]$

$R\_IFromJ[i,j,x] \leftarrow$ symmetric to $L\_JFromI[i,j,x]$