DSI: An Evidence-Based Approach to Identify Dynamic Data Structures in C Programs

Appendix A
Detailed Information for Memory Structures

In this section we give detailed information for each memory structure in scope, covering the following points:

- a *unique ID* that serves as the evidence label;
- an *illustrative points-to graph* to give a flavor of the structure recognized;
- a *visualization of the area condition* associated with the illustrative points-to graph;
- areaCondition, which determines the area of the strand graph potentially containing the memory structure;
- shapePredicate, which confirms an observation of a memory structure by performing a detailed analysis of the area from areaCondition;
- assignEvidence, which, on successful recognition of the memory structure, imparts suitable evidence onto the SCs in the strand graph.

In the illustrative points-to graphs, cells are drawn as circles and strands as block arrows; note that our illustrations do not depict all corner cases. A connection between cells by overlay is represented by a box enclosing both cells. Each box should not be understood as a unique memory region; it is quite acceptable for multiple boxes to reside in the same memory region. Connections between cells that correspond to indirect SCs are simply represented by pointers.

To simplify the presentation, we adopt a style of pattern matching that additionally binds variables on their first appearance. Consider the following predicate:

$$\exists S_p \in \mathcal{S}_t, \mathcal{C} \subseteq \mathcal{C}_t^{\text{rem}}, w, x \in \mathbb{N} :$$
$$\wedge \forall (S_p \xleftrightarrow{xw} S_c) \in \mathcal{C}_t^{\text{rem}} : \ldots$$

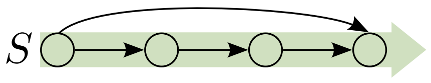
In the $\forall$ quantifier, S_p, x and w are already bound and thus a form of pattern matching takes place over the elements of $\mathcal{C}_t^{\text{rem}}$. The currently unbound variable S_c becomes bound as a result of the pattern match.

It is often convenient for the shape predicate to refer to graph elements bound in the area condition. To avoid writing all output parameters of areaCondition which are subsequently used by the shape predicate, we highlight bound variables that are exported in red and their subsequent usages in the other predicates in blue. This binding process can be seen in the above example predicate.

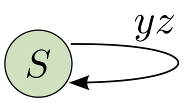
We discuss in the main body of the paper how the evidence weight is derived from the shape predicate, so in the following we only remark on the non-obvious cases.

ID: Tail Pointer - Indirect (TP$_\text{I}$)

A list with a pointer from the head node to the tail node. This memory structure considers the case where the tail pointer does not form a strand in our abstraction, hence the SC to be analyzed is indirect and appears as a loop on the list strand. See also: TP$_\text{O}$.



Points-to Graph



Area Condition

```
AREACONDITION($C_\text{init}, \mathcal{C}_t^\text{rem}, \mathcal{S}_t, \mathcal{C}_t$) =
    if $C_\text{init} = S \xrightarrow{yz} S$
    then return $\{C_\text{init}\}$
    else return $\emptyset$
```

SHAPEPREDICATE =

$$S^{\circlearrowleft} = \emptyset \implies \text{PAIRS}(S \xrightarrow{yz} S) = \{(S^{\rightarrow}[1], S^{\rightarrow}[\text{LENGTH}(S)])\}$$
$$\wedge\ S^{\rightarrow} = \emptyset \implies \exists S_c^{\circlearrowleft} \in \text{CYCLICPERMUATIONS}(S^{\circlearrowleft}) : \text{PAIRS}(S \xrightarrow{yz} S) = \{(S_c^{\circlearrowleft}[1], S_c^{\circlearrowleft}[\text{LENGTH}(S)])\}$$
$$\wedge\ \neg(S^{\rightarrow} \neq \emptyset \wedge S^{\circlearrowleft} \neq \emptyset)$$

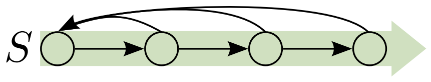
Remarks: If S is allowed to be cyclic, we must check all cyclic permutations. It is not permitted that the list has both a linear and cyclic portion, i.e., a lasso.

ASSIGNEVIDENCE(G_t^s) =

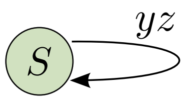
$S \xrightarrow{yz} S$ (in G_t^s) $\leftarrow$ “TP$_\text{I}$” : 3

ID: Head Pointers - Indirect ($\mathrm{HP_I}$)

A list with pointers from all non-head elements back to the head node. The head node may optionally have a pointer to itself. This memory structure considers the case where each head pointer does not form a strand in our abstraction, hence the SC to be analyzed is indirect and appears as a loop on the list strand. See also: $\mathrm{HP_O}$.



Points-to Graph



Area Condition

```
AREACONDITION($C_{\text{init}}, \mathcal{C}_t^{\text{rem}}, \mathcal{S}_t, \mathcal{C}_t$) =
    if $C_{\text{init}} = S \xrightarrow{yz} S$
    then return $\{C_{\text{init}}\}$
    else return $\emptyset$
```

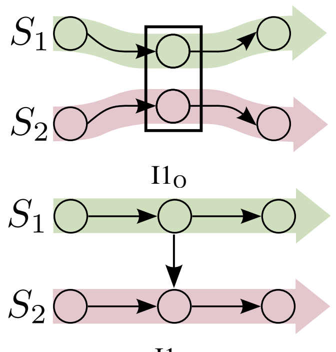
```
SHAPEPREDICATE =
        $S^{\circlearrowleft} = \emptyset \implies$
            $\forall i = 2..\text{LENGTH}(S)$ :
                $(S^{\rightarrow}[i], S^{\rightarrow}[1]) \in \text{PAIRS}(S \xrightarrow{yz} S)$
    $\wedge\ S^{\rightarrow} = \emptyset \implies$
            $\exists S_c^{\circlearrowleft} \in \text{CYCLICPERMUATIONS}(S^{\circlearrowleft})$ :
                $\forall i = 2..\text{LENGTH}(S)$ :
                    $(S_c^{\circlearrowleft}[i], S_c^{\circlearrowleft}[1]) \in \text{PAIRS}(S \xrightarrow{yz} S)$
    $\wedge\ \neg(S^{\rightarrow} \neq \emptyset \wedge S^{\circlearrowleft} \neq \emptyset)$
```

Remarks: If S is allowed to be cyclic, we must check all cyclic permutations. It is not permitted that the list has both a linear and cyclic portion, i.e., a lasso.

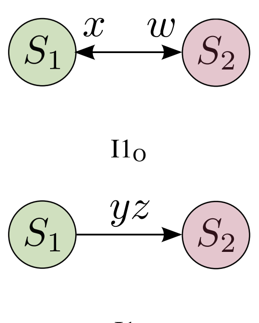
```
ASSIGNEVIDENCE($G_t^s$) =
    $S \xrightarrow{yz} S$ (in $G_t^s$) $\leftarrow$ "$\mathrm{HP_I}$" : $|\text{PAIRS}(S \xrightarrow{yz} S)| * 3$
```

ID: Intersecting Lists on 1 Node - Overlay/Indirect ($I1_O$/$I1_I$)

Two lists that intersect on one node either by overlay, i.e., there exists a memory chunk containing one cell of each list, or by indirection, i.e., there exists a pointer from a cell of one list to a cell of the other. For intersecting lists on one node with an *indirect connection*, replace all instances of $\overset{xw}{\longleftrightarrow}$ with $\overset{yz}{\longrightarrow}$ and "$I1_O$" with "$I1_I$". Replacement sites are marked with gray background.



$I1_I$
Points-to Graphs



$I1_I$
Area Conditions

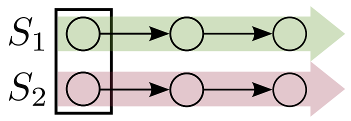
```
AREACONDITION($C_{\text{init}}, \mathcal{C}_t^{\text{rem}}, \mathcal{S}_t, \mathcal{C}_t$) =
    if $C_{\text{init}} = S_1 \overset{xw}{\longleftrightarrow} S_2 \wedge S_1 \neq S_2$
    then return $\{C_{\text{init}}\}$
    else return $\emptyset$
```

SHAPEPREDICATE $= |\text{PAIRS}(S_1 \overset{xw}{\longleftrightarrow} S_2)| = 1$

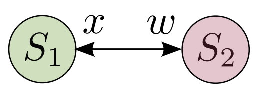
```
ASSIGNEVIDENCE($G_t^s$) =
    $S_1 \overset{xw}{\longleftrightarrow} S_2$ (in $G_t^s$) $\leftarrow$ "$I1_O$" : 1
```

ID: Intersecting Lists on 1 Node - "Same Head Node" (SHN)

A special case of $I1_O$ where the head nodes of two lists reside in the same memory chunk.



Points-to Graph



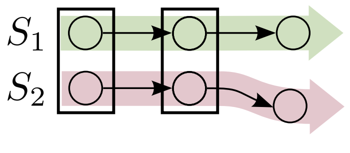
Area Condition

```
AREACONDITION($C_{\text{init}}, \mathcal{C}_t^{\text{rem}}, \mathcal{S}_t, \mathcal{C}_t$) =
    if $C_{\text{init}} = S_1 \overset{xw}{\longleftrightarrow} S_2$
    then return $\{C_{\text{init}}\}$
    else return $\emptyset$
```

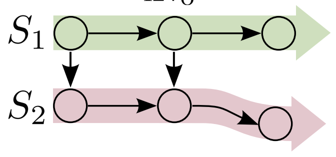
SHAPEPREDICATE $= S_1^{\circlearrowleft} = \emptyset \wedge S_2^{\circlearrowleft} = \emptyset \wedge \text{PAIRS}(S_1 \overset{xw}{\longleftrightarrow} S_2) = \{(S_1^{\rightarrow}[1], S_2^{\rightarrow}[1])\}$

```
ASSIGNEVIDENCE($G_t^s$) =
    $S_1 \overset{xw}{\longleftrightarrow} S_2$ (in $G_t^s$) $\leftarrow$ "SHN" : 3
```

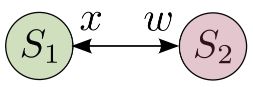
ID: Intersecting Lists on 2+ Nodes - Overlay/Indirect (I2+$_{\mathrm{O}}$/I2+$_{\mathrm{I}}$)
Two lists that intersect on two or more nodes by overlay or indirection. This predicate is typically matched on doubly linked lists and skip lists that currently have a degenerate shape. For intersecting lists on two or more nodes with an *indirect connection*, replace all instances of $\overset{xw}{\longleftrightarrow}$ with $\xrightarrow{yz}$ and "I2+$_{\mathrm{O}}$" with "I2+$_{\mathrm{I}}$". Replacement sites are marked with gray background.



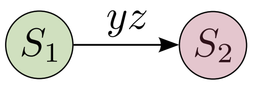
I2+$_{\mathrm{O}}$



I2+$_{\mathrm{I}}$
Points-to Graphs



I2+$_{\mathrm{O}}$



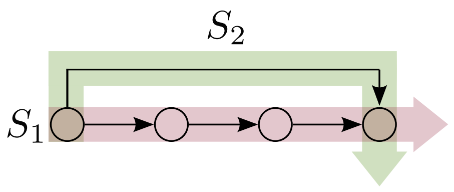
I2+$_{\mathrm{I}}$
Area Conditions

```
AREACONDITION($C_{\text{init}}, \mathcal{C}_t^{\text{rem}}, \mathcal{S}_t, \mathcal{C}_t$) =
    if $C_{\text{init}} = S_1 \overset{xw}{\longleftrightarrow} S_2 \wedge S_1 \neq S_2$
    then return $\{C_{\text{init}}\}$
    else return $\emptyset$
```

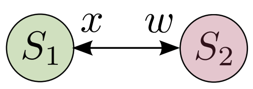
SHAPEPREDICATE $= |\text{PAIRS}(S_1 \overset{xw}{\longleftrightarrow} S_2)| > 1$

ASSIGNEVIDENCE(G_t^s) =
$S_1 \overset{xw}{\longleftrightarrow} S_2$ (in G_t^s) $\leftarrow$ "I2+$_{\mathrm{O}}$" : $|\text{PAIRS}(S_1 \overset{xw}{\longleftrightarrow} S_2)|$

ID: Tail Pointer - Overlay ($\overrightarrow{\text{TP}}_\text{O}$)
A list with a pointer from the head node to the tail node. This memory structure considers the case where the tail pointer forms a strand in our abstraction. See also: TP_I.



Points-to Graph



Area Condition

```
AREACONDITION($C_\text{init}, \mathcal{C}_t^\text{rem}, \mathcal{S}_t, \mathcal{C}_t$) =
  if $C_\text{init} = S_1 \overset{ab}{\longleftrightarrow} S_2$
  then return $\{C_\text{init}\}$
  else return $\emptyset$
```

$\text{SHAPEPREDICATE} = \exists(S_\text{list}, S_\text{tail}, x, w) \in \{(S_1, S_2, a, b), (S_2, S_1, b, a)\}$:

$$\text{LENGTH}(S_\text{tail}) = 2 \wedge S_\text{tail}^{\circlearrowleft} = \emptyset$$
$$\wedge\ S_\text{list}^{\circlearrowleft} = \emptyset \implies |\text{PAIRS}(S_\text{list} \overset{xw}{\longleftrightarrow} S_\text{tail})| = \{(S_\text{list}^{\rightarrow}[1], S_\text{tail}^{\rightarrow}[1]), (S_\text{list}^{\rightarrow}[\text{LENGTH}(S_\text{list})], S_\text{tail}^{\rightarrow}[2])\}$$
$$\wedge\ S_\text{list}^{\rightarrow} = \emptyset \implies S_\text{clist}^{\circlearrowleft} \in \text{CYCLICPERMUATIONS}(S_\text{list}^{\circlearrowleft}) :$$
$$|\text{PAIRS}(S_\text{list} \overset{xw}{\longleftrightarrow} S_\text{tail})| = \{(S_\text{clist}^{\circlearrowleft}[1], S_\text{tail}^{\rightarrow}[1]), (S_\text{clist}^{\circlearrowleft}[\text{LENGTH}(S_\text{list})], S_\text{tail}^{\rightarrow}[2])\}$$
$$\wedge\ \neg(S_\text{list}^{\rightarrow} \neq \emptyset \wedge S_\text{list}^{\circlearrowleft} \neq \emptyset)$$

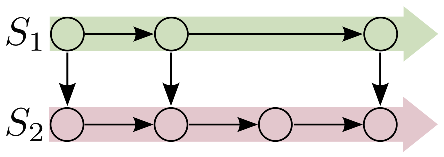
Remarks: S_tail must be exactly of length two and non-cyclic. If S_list is cyclic, then a cyclic permutation of the cell sequence must be found to match the cell sequence of the tail strand. It is not permitted that S_list has both a linear and cyclic portion, i.e., a lasso.

$\text{ASSIGNEVIDENCE}(G_t^s) =$
$S_\text{list} \overset{xw}{\longleftrightarrow} S_\text{tail}$ (in G_t^s) $\leftarrow$ "$\overleftarrow{\text{TP}}_\text{O}$" : 6

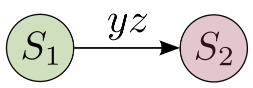
Remarks: Label direction ($\overrightarrow{\text{TP}}_\text{O}$) points from tail strand to list strand.

ID: Skip List - Indirect (SL_I)

A two-level skip list where the downward connection between the upper (S_high) and lower (S_low) levels is made by an indirect SC. This situation arises when the downward pointers do not form strands (see $\vec{\text{SL}}_{\text{O2}}$ for the case when the downward pointers do form strands). Typically there exists a chain of discovered two-level skip-lists that will be combined back into one multi-level skip list in the naming phase. See also: $\vec{\text{SL}}_{\text{O1}}$ and $\vec{\text{SL}}_{\text{O2}}$.



Points-to Graph



Area Condition

```
AREACONDITION$(C_\text{init}, \mathcal{C}_t^\text{rem}, \mathcal{S}_t, \mathcal{C}_t)$ =
    if $C_\text{init} = S_\text{high} \xrightarrow{yz} S_\text{low} \wedge S_\text{high} \neq S_\text{low}$
    then return $\{C_\text{init}\}$
    else return $\emptyset$
```

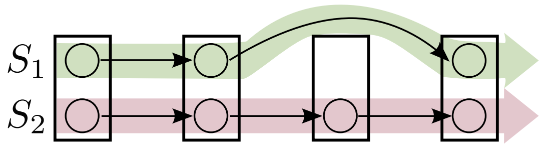
```
SHAPEPREDICATE =
      $\nexists(S_\text{high} \overset{kk}{\longleftrightarrow} S_\text{low}) \in \mathcal{C}_t$
    $\wedge\, S_\text{high}^{\circlearrowleft} = \emptyset \wedge S_\text{low}^{\circlearrowleft} = \emptyset$
    $\wedge\, \forall i \in [1, \text{LENGTH}(S_\text{high}) - 1] \exists j, j' \in [1, \text{LENGTH}(S_\text{low})]$ :
          $(S_\text{high}^{\rightarrow}[i], S_\text{low}^{\rightarrow}[j]) \in \text{PAIRS}(S_\text{high} \xrightarrow{yz} S_\text{low})$
        $\wedge\, (S_\text{high}^{\rightarrow}[i+1], S_\text{low}^{\rightarrow}[j']) \in \text{PAIRS}(S_\text{high} \xrightarrow{yz} S_\text{low})$
        $\wedge\, j < j'$
```

Remarks: There must exist no sharing between the two skip list levels and we currently exclude cyclic skip lists (although this could be handled by finding a suitable cyclic permutation of the cell sequence for one of the levels). The key shape property to be checked is that when adjacent cells from the upper level ($S_\text{high}^{\rightarrow}[i]$ & $S_\text{high}^{\rightarrow}[i+1]$) are mapped to those on the lower level ($S_\text{low}^{\rightarrow}[j]$ & $S_\text{low}^{\rightarrow}[j']$), the relative ordering of those target cells should match that of the upper level, i.e., $j < j'$.

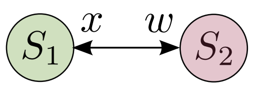
```
ASSIGNEVIDENCE$(G_t^s)$ =
    $S_\text{high} \xrightarrow{yz} S_\text{low}$ (in $G_t^s$) $\leftarrow$ "$\text{SL}_\text{I}$" : $|\text{PAIRS}(S_\text{high} \xrightarrow{yz} S_\text{low})| * 3$
```

ID: Skip List - Overlay in struct ($\vec{\text{SL}}_{\text{O1}}$)

A two-level skip list where the downward connection between the upper (S_{high}) and lower (S_{low}) levels is made by an overlay SC. This situation typically arises when there exists a struct holding a cell of each level of the skip list. Typically there exists a chain of discovered two-level skip-lists that will be combined back into one multi-level skip list in the naming phase. See also: SL_{I} and $\vec{\text{SL}}_{\text{O2}}$.



Points-to Graph



Area Condition

```
AREACONDITION($C_{\text{init}}, \mathcal{C}_t^{\text{rem}}, \mathcal{S}_t, \mathcal{C}_t$) =
    if $C_{\text{init}} = S_1 \overset{ab}{\longleftrightarrow} S_2$
    then return $\{C_{\text{init}}\}$
    else return $\emptyset$
```

```
SHAPEPREDICATE =
        $\nexists(S_1 \overset{kk}{\longleftrightarrow} S_2) \in \mathcal{C}_t$
    $\wedge\ S_1^{\circlearrowleft} = \emptyset \wedge S_2^{\circlearrowleft} = \emptyset$
    $\wedge\ \exists(S_{\text{high}}, S_{\text{low}}, x, w) \in \{(S_1, S_2, a, b), (S_2, S_1, b, a)\}$ :
        $\forall i \in [1, \text{LENGTH}(S_{\text{high}}) - 1] \exists j, j' \in [1, \text{LENGTH}(S_{\text{low}})]$ :
            $(S_{\text{high}}^{\rightarrow}[i], S_{\text{low}}^{\rightarrow}[j]) \in \text{PAIRS}(S_{\text{high}} \overset{xw}{\longleftrightarrow} S_{\text{low}})$
          $\wedge\ (S_{\text{high}}^{\rightarrow}[i+1], S_{\text{low}}^{\rightarrow}[j']) \in \text{PAIRS}(S_{\text{high}} \overset{xw}{\longleftrightarrow} S_{\text{low}})$
          $\wedge\ j < j'$
```

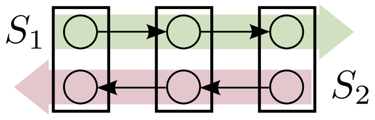
Remarks: The shape predicate is very similar to that for SL_{I}, however, as a deep inspection of the strands is necessary to establish which is the upper/lower level, this choice must be delayed until inside the shape predicate.

```
ASSIGNEVIDENCE($G_t^s$) =
    $S_{\text{high}} \overset{xw}{\longleftrightarrow} S_{\text{low}}$ (in $G_t^s$) $\leftarrow$ "$\vec{\text{SL}}_{\text{O1}}$" : $|\text{PAIRS}(S_{\text{high}} \overset{xw}{\longleftrightarrow} S_{\text{low}})| * 3$
```

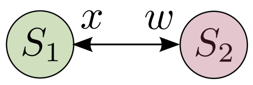
Remarks: Label direction ($\vec{\text{SL}}_{\text{O1}}$) points from higher level to lower level.

ID: Doubly Linked List - Overlay (DLL)

A (non-cyclic) doubly linked list where the connection between the forward and reverse direction is formed by an overlay SC. Although it would be trivial to include a DLL where the connection is formed by an indirect SC, this does not appear in practice. See also: CDLL.



Points-to Graph



Area Condition

```
AREACONDITION($C_{\text{init}}, \mathcal{C}_t^{\text{rem}}, \mathcal{S}_t, \mathcal{C}_t$) =
    if $C_{\text{init}} = S_1 \overset{xw}{\longleftrightarrow} S_2$
    then return $\{C_{\text{init}}\}$
    else return $\emptyset$
```

SHAPEPREDICATE =

$$S_1^{\circlearrowleft} = \emptyset \land S_2^{\circlearrowleft} = \emptyset$$
$$\land \text{ LENGTH}(S_1) = \text{LENGTH}(S_2) = |\text{PAIRS}(S_1 \overset{xw}{\longleftrightarrow} S_2)|$$
$$\land \ \forall i \in [0..\text{LENGTH}(S_1) - 1] \ \exists (c_1, c_2) \in \text{PAIRS}(S_1 \overset{xw}{\longleftrightarrow} S_2) :$$
$$S_1^{\rightarrow}[i+1] = c_1 \land S_2^{\rightarrow}[\text{LENGTH}(S_2) - i] = c_2$$

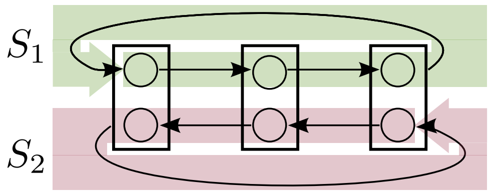
Remarks: We must check that for every cell in the forward direction $S_1^{\rightarrow}[i+1]$ there exists a connection to the appropriate cell in the reverse direction $S_1^{\rightarrow}[\text{LENGTH}(S_2) - i]$. The choice of direction is arbitrary at this point.

ASSIGNEVIDENCE(G_t^s) =

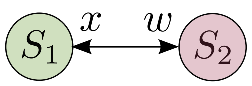
$S_1 \overset{xw}{\longleftrightarrow} S_2$ (in G_t^s) $\leftarrow$ "DLL" : $|\text{PAIRS}(S_1 \overset{xw}{\longleftrightarrow} S_2)| * 3$

ID: Cyclic Doubly Linked List - Overlay (CDLL)

A cyclic doubly linked list as exemplified by the Linux kernel list. See also: DLL.



Points-to Graph



Area Condition

$\textsc{areaCondition}(C_{\text{init}}, \mathcal{C}_t^{\text{rem}}, \mathcal{S}_t, \mathcal{C}_t) =$

```
    if $C_{\text{init}} = S_1 \overset{xw}{\longleftrightarrow} S_2$
    then return $\{C_{\text{init}}\}$
    else return $\emptyset$
```

$\textsc{shapePredicate} =$

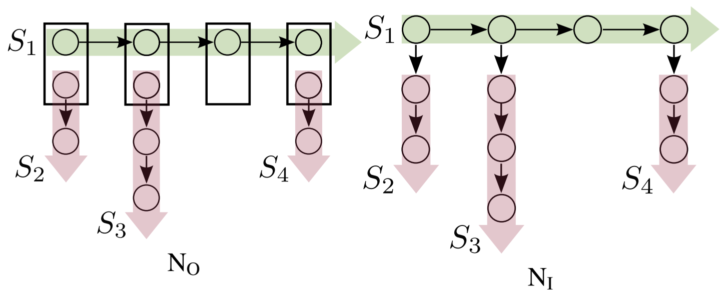
$$S_1^{\rightarrow} = \emptyset \land S_2^{\rightarrow} = \emptyset$$
$$\land\ \textsc{length}(S_1) = \textsc{length}(S_2) = |\textsc{pairs}(S_1 \overset{xw}{\longleftrightarrow} S_2)|$$
$$\land\ \exists S_{1c}^{\circlearrowleft} \in \textsc{cyclicPermuations}(S_1^{\circlearrowleft}) :$$
$$\forall i \in [0..\textsc{length}(S_1) - 1]\ \exists (c_1, c_2) \in \textsc{pairs}(S_1 \overset{xw}{\longleftrightarrow} S_2) :$$
$$S_{1c}^{\circlearrowleft}[i+1] = c_1 \land S_2^{\circlearrowleft}[\textsc{length}(S_2) - i] = c_2$$

Remarks: Similar to the shape predicate for DLL, but an additional cyclic permutation for the cell sequence of one list must be found to place the cell pairs in alignment.

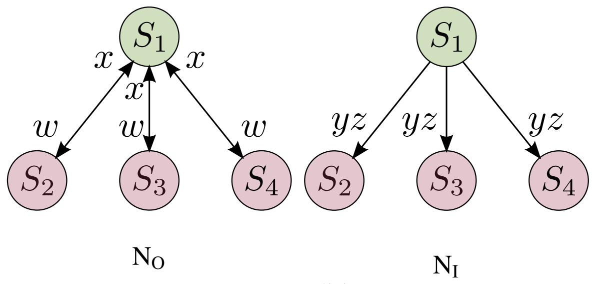
$\textsc{assignEvidence}(G_t^s) =$

$$S_1 \overset{xw}{\longleftrightarrow} S_2 \text{ (in } G_t^s) \leftarrow \text{“CDLL”} : |\textsc{pairs}(S_1 \overset{xw}{\longleftrightarrow} S_2)| * 3$$

ID: Nesting - Overlay/Indirect ($\vec{\mathrm{N}}_{\mathrm{O}}$/$\mathrm{N}_{\mathrm{I}}$)
Nesting between data structures via a parent-child relationship. The connection from parent to child may take place by overlay, i.e., when a cell of the parent and child share the same memory chunk, or by indirection, i.e., when there exists a pointer from the memory chunk of the parent cell to the memory chunk of the child cell. This predicate does not distinguish between unique/shared children and the connection point to the child (i.e., to the head node of the child). We believe all these specializations can be handled by our approach, but an exhaustive enumeration remains future work. To convert the following from *Nesting - Overlay* to *Nesting - Indirect*, replace all instances of $\overset{xw}{\longleftrightarrow}$ with $\overset{yz}{\longrightarrow}$ and "$\vec{\mathrm{N}}_{\mathrm{O}}$" with "$\mathrm{N}_{\mathrm{I}}$". Replacement sites are marked with a gray background.



Points-to Graphs



Area Conditions

```
AREACONDITION($C_{\text{init}}, \mathcal{C}_t^{\text{rem}}, \mathcal{S}_t, \mathcal{C}_t$) =
  if $\exists S_p \in \mathcal{S}_t, \mathcal{S}_c \subset \mathcal{S}_t, \mathcal{C} \subseteq \mathcal{C}_t^{\text{rem}}, x, w \in \mathbb{N}$ :
      $C_{\text{init}} \in \mathcal{C}$                                                          ▷ Must include $C_{\text{init}}$
     $\wedge\forall(S_p \overset{xw}{\longleftrightarrow} S_c) \in \mathcal{C}_t^{\text{rem}} : (S_p \overset{xw}{\longleftrightarrow} S_c) \in \mathcal{C} \wedge S_c \in \mathcal{S}_c \wedge S_p \neq S_c$      ▷ Ensure maximal subgraph is found
     $\wedge\forall S_c \in \mathcal{S}_c : (S_p \overset{xw}{\longleftrightarrow} S_c) \in \mathcal{C}$                                   ▷ No unwanted strands
     $\wedge|\mathcal{C}| = |\mathcal{S}_c|$                                                        ▷ No unwanted SCs
     $\wedge|\mathcal{S}_c| \geq 2$
  then return $\mathcal{C}$
  else return $\emptyset$
```

Remarks: Typically there should exist a unique S_p satisfying the area condition, however, when this is not the case then the shape is likely a degenerate tree or skip list. Thus, it does not matter if some temporally arbitrarily orientated nesting evidence is gathered as this will be overridden by the correct data structure during the naming phase.

SHAPEPREDICATE = ALLLINKAGECONDITIONSEQUAL($\mathcal{S}_c$)

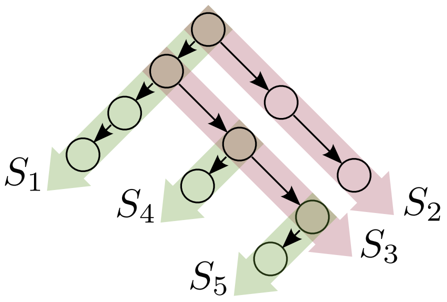
Remarks: The area predicate cannot look inside the child strands and check that they all have the same form, thus this is done in the shape predicate. Note that $\vec{\mathrm{N}}_{\mathrm{O}}$.SHAPEPREDICATE is this simple, i.e., it only checks that all children have the same linkage condition, because many other possibilities have already been filtered out due to the priority system over memory structures, i.e., trees and skip lists have already failed to match. Thus, nesting becomes the only possible interpretation.

```
ASSIGNEVIDENCE($G_t^s$) =
for all $S_c \in \mathcal{S}_c$ do
   $S_p \overset{xw}{\longleftrightarrow} S_c$ (in $G_t^s$) ← "$\vec{\mathrm{N}}_{\mathrm{O}}$" : 1
```

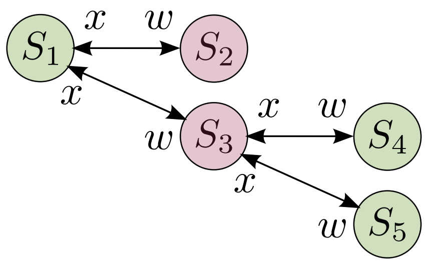
Remarks: This shape predicate does not investigate the cell pairs of the SCs via PAIRS. Thus, it is the only memory structure where the weight of evidence is not directly derived from the shape predicate. Instead, by construction, there must be at least one pair in the cell-pairs relationship for each SC, and as we do not inspect this further, the evidence is simply 1 per SC.
Label direction ($\vec{\mathrm{N}}_{\mathrm{O}}$) points from parent strand to child strand.

ID: Binary Tree - Overlay (BT)

A binary tree where the connection between the lists forming the left branches and the lists forming the right branches is made by overlay SCs. If the connections are indirect SCs then the data structure is considered $\mathrm{N_I}$.



Points-to Graph



Area Condition

```
AREACONDITION($C_{\text{init}}, \mathcal{C}_t^{\text{rem}}, \mathcal{S}_t, \mathcal{C}_t$) =
  if $\exists \mathcal{S}_l \subset \mathcal{S}_t, \mathcal{S}_r \subset \mathcal{S}_t, \mathcal{C} \subseteq \mathcal{C}_t^{\text{rem}}, x, w \in \mathbb{N}$ :
      $C_{\text{init}} \in \mathcal{C}$                                                    ▷ Must include $C_{\text{init}}$
      $\wedge \forall S_1 \in \mathcal{S}_l \forall (S_1 \overset{xw}{\longleftrightarrow} S_2) \in \mathcal{C}_t^{\text{rem}} : (S_1 \overset{xw}{\longleftrightarrow} S_2) \in \mathcal{C} \wedge S_2 \in \mathcal{S}_r$     ▷ Ensure maximal subgraph is found
      $\wedge \forall S_1 \in \mathcal{S}_r \forall (S_1 \overset{wx}{\longleftrightarrow} S_2) \in \mathcal{C}_t^{\text{rem}} : (S_1 \overset{wx}{\longleftrightarrow} S_2) \in \mathcal{C} \wedge S_2 \in \mathcal{S}_l$       with alternating structure
      $\wedge \forall (S_1 \overset{\alpha}{\cdots} S_2) \in \mathcal{C}$ :
          $\overset{\alpha}{\cdots} = \longleftrightarrow$                                           ▷ All SCs are overlays
        $\wedge (S_1 \overset{\alpha}{\cdots} S_2 = S_1 \overset{xw}{\longleftrightarrow} S_2 \implies S_1 \in \mathcal{S}_l \wedge S_2 \in \mathcal{S}_r)$            ▷ No unwanted SCs
        $\wedge (S_1 \overset{\alpha}{\cdots} S_2 = S_1 \overset{wx}{\longleftrightarrow} S_2 \implies S_1 \in \mathcal{S}_r \wedge S_2 \in \mathcal{S}_l)$
      $\wedge \forall (S_1, S_2) \in (\mathcal{S}_l \cup \mathcal{S}_r) \times (\mathcal{S}_l \cup \mathcal{S}_r) : \nexists (S_1 \overset{kk}{\longleftrightarrow} S_2) \in \mathcal{C}_t$ ▷ No sharing between strands of subgraph (checked in $\mathcal{C}_t$)
      $\wedge$CONNECTEDGRAPH($\mathcal{S}_l \cup \mathcal{S}_r, \mathcal{C}$)
      $\wedge \exists path \in (\mathcal{S}_l \cup \mathcal{S}_r, \mathcal{C})$ : LEN($path$) $> 3 \wedge path$ visits each vertex at most once     ▷ Differentiate BT from $\vec{\mathrm{N}}_{\mathrm{O}}$
      $\wedge$ACYCLICGRAPH($\mathcal{S}_l \cup \mathcal{S}_r, \mathcal{C}$)                          ▷ Differentiate BT from $\vec{\mathrm{SL}}_{\mathrm{O2}}$
  then return $\mathcal{C}$
  else return $\emptyset$
```

Remark: The key property to test is the alternating structure of the tree. BT and $\vec{\mathrm{N}}_{\mathrm{O}}$ have inherent similarity; this is resolved by requiring a path passing through unique vertices of at least length 4. A distinction to the area condition of $\vec{\mathrm{SL}}_{\mathrm{O2}}$ is made by requiring the subgraph to be acyclic.

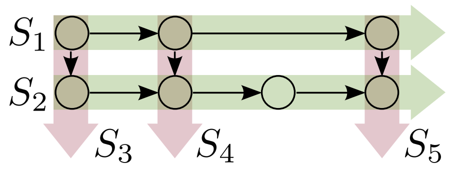
```
SHAPEPREDICATE =
    NOCYCLICSTRANDS($\mathcal{S}_l \cup \mathcal{S}_r$)
  $\wedge$ ALLLINKAGECONDITIONSEQUAL($\mathcal{S}_l$)
  $\wedge$ ALLLINKAGECONDITIONSEQUAL($\mathcal{S}_r$)
  $\wedge \exists S_{\text{root}} \in \mathcal{S}_l \cup \mathcal{S}_r$ :
      $S_{\text{root}} \in \mathcal{S}_l \implies$ TREEAUX($S_{\text{root}}, x, w, \mathcal{S}_r, \mathcal{S}_l$)
      $S_{\text{root}} \in \mathcal{S}_r \implies$ TREEAUX($S_{\text{root}}, w, x, \mathcal{S}_l, \mathcal{S}_r$)
where
  TREEAUX($S_{\text{root}}, a, b, \mathcal{S}_a, \mathcal{S}_b$) =
    $\forall S_a \in \mathcal{S}_a : \exists (S_{\text{root}} \overset{ab}{\longleftrightarrow} S_a) \in \mathcal{C} \implies$
        (PAIRS($S_{\text{root}} \overset{ab}{\longleftrightarrow} S_a$) $= \{(\_, S_a^{\rightarrow}[1])\}$
      $\wedge$ TREEAUX($S_a, b, a, \mathcal{S}_b, \mathcal{S}_a$))
```

Remarks: Once a root strand S_{root} is found, the alternating structure of the tree is confirmed via the recursive predicate TREEAUX. On each call, the SC from the current root to each child is checked to ensure that the target of the SC is the first cell of the child ($S_a^{\rightarrow}[1]$).

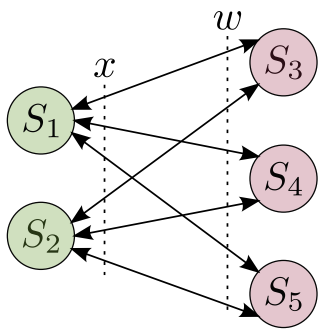
```
ASSIGNEVIDENCE($G_t^s$) =
  for all $(S_a \overset{ab}{\longleftrightarrow} S_b) \in \mathcal{C}$ do
    $S_a \overset{ab}{\longleftrightarrow} S_b$ (in $G_t^s$) $\leftarrow$ "BT" : 2
```

Remarks: Note that the shape predicate disregards the position of the parent cell when examining a cell pair, thus the weight of evidence is checking for the existence of the atomic cell pair plus examining the position of the child cell, i.e., two per SC.

ID: Skip List - Overlay ($\vec{\mathrm{SL}}_{\mathrm{O2}}$)
A skip list where the downward links appear as strands, i.e., they are composed of pointers which fulfill minimum conditions. This implementation leads to a skip list that can easily be extended by the addition of extra horizontal levels. See also: SL_I and $\vec{\mathrm{SL}}_{\mathrm{O1}}$.



Points-to Graph



Area Condition

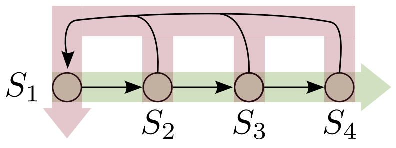
```
AREACONDITION($C_{\text{init}}, \mathcal{C}_t^{\text{rem}}, \mathcal{S}_t, \mathcal{C}_t$) =
    Identical to BT.AREACONDITION, except $\mathcal{S}_1 = \mathcal{S}_l$ and $\mathcal{S}_2 = \mathcal{S}_r$. Furthermore, CYCLICGRAPH($\mathcal{S}_1 \cup \mathcal{S}_2, \mathcal{C}$) is required instead of ACYCLICGRAPH($\mathcal{S}_1 \cup \mathcal{S}_2, \mathcal{C}$)

SHAPEPREDICATE =
      NOCYCLICSTRANDS($\mathcal{S}_1 \cup \mathcal{S}_2$)
    $\wedge$ ALLLINKAGECONDITIONSEQUAL($\mathcal{S}_1$)
    $\wedge$ ALLLINKAGECONDITIONSEQUAL($\mathcal{S}_2$)
    $\wedge \exists(\mathcal{S}_\mathrm{h}, \mathcal{S}_\mathrm{v}, x, w) \in \{(\mathcal{S}_1, \mathcal{S}_2, a, b), (\mathcal{S}_2, \mathcal{S}_1, b, a)\}$ :
        $\exists\langle H_1, H_2, \ldots, H_{|\mathcal{S}_h|}\rangle \in$ SEQUENCES($\mathcal{S}_h$) :
            $\forall i \in [1, |\mathcal{S}_h| - 1]$ :                              ▷ Condition 1
                $\forall j \in [1,$ LENGTH$(H_i) - 1]$ :
                    $\exists S_v, S_{v'} \in \mathcal{S}_v, a, b, c, c' \in \mathbb{N}$ :
                          $\{(H_i^{\rightarrow}[j], S_v^{\rightarrow}[a])\} \in$ PAIRS$(H_i \overset{xw}{\longleftrightarrow} S_v)$
                        $\wedge \{(H_i^{\rightarrow}[j+1], S_{v'}^{\rightarrow}[b])\} \in$ PAIRS$(H_i \overset{xw}{\longleftrightarrow} S_{v'})$
                        $\wedge \{(H_{i+1}^{\rightarrow}[c], S_v^{\rightarrow}[a+1])\} \in$ PAIRS$(H_{i+1} \overset{xw}{\longleftrightarrow} S_v)$
                        $\wedge \{(H_{i+1}^{\rightarrow}[c'], S_{v'}^{\rightarrow}[b+1])\} \in$ PAIRS$(H_{i+1} \overset{xw}{\longleftrightarrow} S_{v'})$
                        $\wedge c < c'$
        $\wedge \forall S_v \in \mathcal{S}_v$ :                                    ▷ Condition 2
            $\forall i \in [1, |\mathcal{S}_v|]$ :
                $(\_, S_v^{\rightarrow}[i]) \in$ PAIRS$(H_{|\mathcal{S}_h| - \mathrm{LENGTH}(S_v) + i} \overset{xw}{\longleftrightarrow} S_v)$
```

Remarks: As with BT.SHAPEPREDICATE, we require no cyclic strands and that the linkage conditions within each partition of strands be equal. We then choose which partition of strands represents the horizontal strands $\mathcal{S}_h$ and which represents the vertical strands $\mathcal{S}_v$ of the skip list. An order $\langle H_1, H_2, \ldots, H_{|\mathcal{S}_h|}\rangle$ over the horizontal strands is set from the most shallow (H_1) to the deepest ($H_{|\mathcal{S}_h|}$). Given this order, two conditions must hold (the start of each is labeled above). The first checks that all adjacent cells in a horizontal strand map to the cells of the next deepest level using the vertical strands, and that those target cells appear in the same horizontal order as the source cells. The second property checks that the sequence of cells appearing in each vertical strand respects the ordering $\langle H_1, H_2, \ldots, H_{|\mathcal{S}_h|}\rangle$ over horizontal strands.

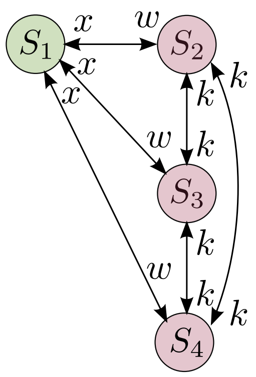
```
ASSIGNEVIDENCE($G_t^s$) =
    for all $(S_a \overset{xw}{\longleftrightarrow} S_b) \in \mathcal{C}$ do
        $S_a \overset{xw}{\longleftrightarrow} S_b$ (in $G_t^s$) ← "$\vec{\mathrm{SL}}_{\mathrm{O2}}$" : 3
```

Remarks: Label direction ($\vec{\mathrm{SL}}_{\mathrm{O2}}$) points from horizontal strand to vertical strand.

ID: Head Pointers - Overlay ($\overrightarrow{\text{HP}}_{\text{O}}$)
A list with pointers from all non-head elements back to the head node. The head node may optionally have a pointer to itself. Each head pointer forms a strand, and is thus connected to the list by an overlay SC. Additionally, since each head pointer strand terminates with the same cell, they all share the same tail cell sequence. This results in the head pointer strands forming a subgraph which is fully connected by SCs describing sharing. See also: HP_{I}.



Points-to Graph



Area Condition

```
AREACONDITION($\mathcal{C}_{\text{init}}, \mathcal{C}_t^{\text{rem}}, \mathcal{S}_t, \mathcal{C}_t$) =
    if $\exists S_{\text{list}} \in \mathcal{S}_t, \mathcal{S}_{\text{head}} \subset \mathcal{S}_t, \mathcal{C} \subseteq \mathcal{C}_t^{\text{rem}}, x, w \in \mathbb{N}$ :
        $C_{\text{init}} \in \mathcal{C}$                                                      ▷ Must include $C_{\text{init}}$
        $\wedge \forall (S_{\text{list}} \overset{xw}{\longleftrightarrow} S_{\text{head}}) \in \mathcal{C}_t^{\text{rem}} : (S_{\text{list}} \overset{xw}{\longleftrightarrow} S_{\text{head}}) \in \mathcal{C} \wedge S_{\text{head}} \in \mathcal{S}_{\text{head}}$      ▷ Ensure maximal subgraph is found
        $\wedge \forall S_{\text{head}} \in \mathcal{S}_{\text{head}} : (S_{\text{list}} \overset{xw}{\longleftrightarrow} S_{\text{head}}) \in \mathcal{C}$                      ▷ No unwanted strands
        $\wedge |\mathcal{C}| = |\mathcal{S}_{\text{head}}| + \binom{|\mathcal{S}_{\text{head}}|+1}{2}$                                  ▷ No unwanted SCs
        $\wedge \forall (S_{\text{head}}, S'_{\text{head}}) \in \mathcal{S}_{\text{head}} \times \mathcal{S}_{\text{head}} : S_{\text{head}} \neq S'_{\text{head}} \implies (S_{\text{head}} \overset{kk}{\longleftrightarrow} S'_{\text{head}}) \in \mathcal{C}$ ▷ Sharing SC between all head pointers
    then return $\mathcal{C}$
    else return $\emptyset$
```

Remarks: The shape predicate functions similarly to that for $\text{N}_{\text{O}}/\text{N}_{\text{I}}$, where the parent-child structure is replicated by the relationship between the main list and each head pointer strand. Additionally, the fully-connected sharing property of the head pointer strands is checked, which results in an extra $\binom{|\mathcal{S}_{\text{head}}|+1}{2}$ SCs in $\mathcal{C}$.

```
SHAPEPREDICATE =
      NOCYCLICSTRANDS($\mathcal{S}_{\text{head}}$)
    $\wedge$ ALLLINKAGECONDITIONSEQUAL($\mathcal{S}_{\text{head}}$)
    $\wedge$ $S_{\text{list}}^{\circlearrowleft} = \emptyset \implies$
          $\forall S_{\text{head}} \in \mathcal{S}_{\text{head}}$ : LENGTH($S_{\text{head}}^{\rightarrow}$) $= 2$
        $\wedge$ $\forall i \in [2,$ LENGTH($S_{\text{list}}^{\rightarrow}$)$] \exists S_{\text{head}} \in \mathcal{S}_{\text{head}}$ :
            $\{S_{\text{list}}^{\rightarrow}[i], S_{\text{head}}^{\rightarrow}[1]), (S_{\text{list}}^{\rightarrow}[1], S_{\text{head}}^{\rightarrow}[2])\}$ = PAIRS($S_{\text{list}} \overset{xw}{\longleftrightarrow} S_{\text{head}}$)
    $\wedge$ $S_{\text{list}}^{\rightarrow} = \emptyset \implies \exists S_{\text{list}}^{\circlearrowleft\text{perm}} \in$ CYCLICPERMUATIONS($S_{\text{list}}^{\circlearrowleft}$) :
          $\forall S_{\text{head}} \in \mathcal{S}_{\text{head}}$ : LENGTH($S_{\text{head}}^{\circlearrowleft}$) $= 2$
        $\wedge$ $\forall i \in [2,$ LENGTH($S_{\text{list}}$)$] \exists S_{\text{head}} \in \mathcal{S}_{\text{head}}$ :
            $\{(S_{\text{list}}^{\circlearrowleft\text{perm}}[i], S_{\text{head}}^{\circlearrowleft}[1]), (S_{\text{list}}^{\circlearrowleft\text{perm}}[1], S_{\text{head}}^{\circlearrowleft}[2])\}$ = PAIRS($S_{\text{list}} \overset{xw}{\longleftrightarrow} S_{\text{head}}$)
    $\wedge$ $\neg(S_{\text{list}}^{\rightarrow} \neq \emptyset \wedge S_{\text{list}}^{\circlearrowleft} \neq \emptyset)$
```

Remarks: In a manner similar to HP_{I}, we perform a case distinction on the main list having a linear or cyclic cell sequence. In the case of a cyclic cell sequence, a cyclic permutation of the main list must be found.

```
ASSIGNEVIDENCE($G_t^s$) =
    for all ($S_{\text{list}} \overset{xw}{\longleftrightarrow} S_{\text{head}}$) $\in \mathcal{C}$ do
        $S_{\text{list}} \overset{xw}{\longleftrightarrow} S_{\text{head}}$ (in $G_t^s$) $\leftarrow$ "$\overleftarrow{\text{HP}}_{\text{O}}$" : 6
```

Remarks: Label direction ($\overrightarrow{\text{HP}}_{\text{O}}$) points from head strand to main list strand.