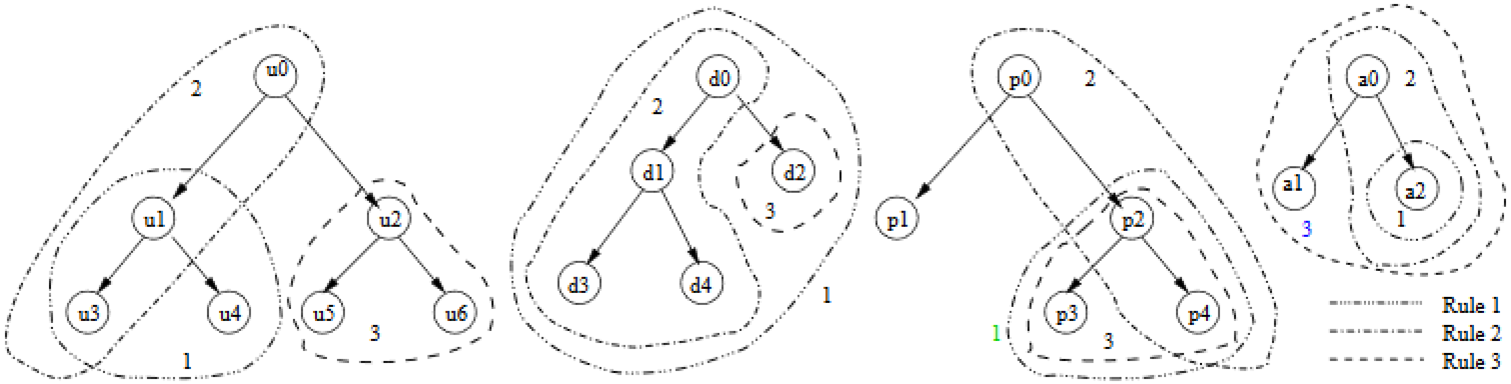
Transformation from the original rule list to the final list of scope-based rules

1. Extend each deny rule to all elements of the hierarchies and then explicitly exclude those elements that must not be affected by this artificially enlarged scope
2. Float down obligation rules (see later how)
3. Float up allow rules until a rule is reached whose scope comprises the scope of the allow rule

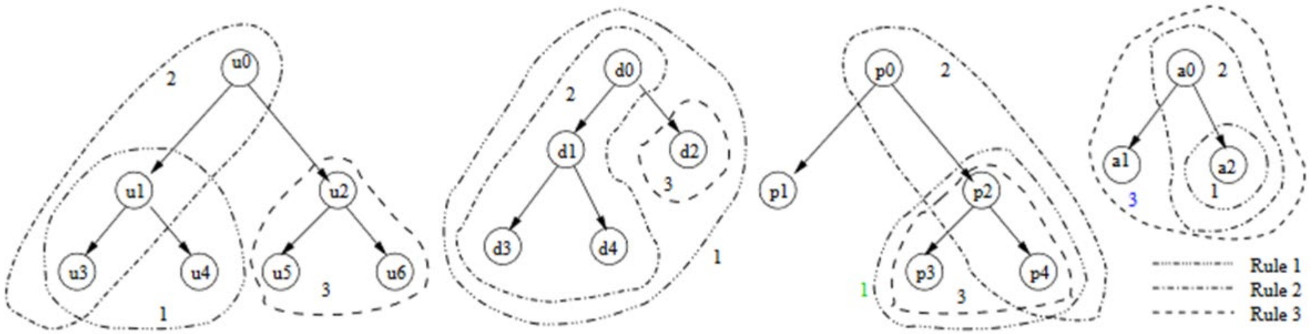
Original rule list & hierarchies



1	$\langle (u_1, d_0, p_2, a_2), (\circ, c_1, \bar{o}_1) \rangle$
2	$\langle (u_3, d_1, p_4, a_2), (-, c_2, \bar{o}_2) \rangle$
3	$\langle (u_2, d_2, p_2, a_0), (+, c_3, \bar{o}_3) \rangle$

1) Deny rules

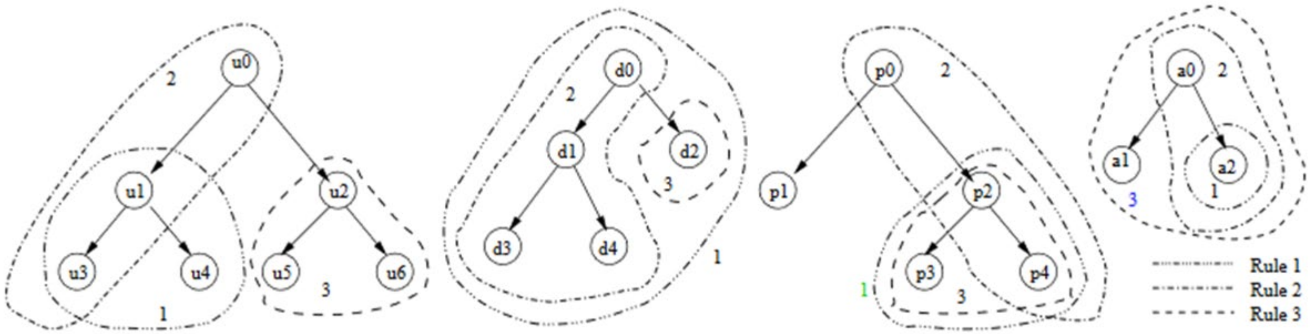
- extend each deny rule to all elements of the hierarchies
- explicitly exclude those elements that must not be affected by this artificially enlarged scope.



1	$\langle (u_1, d_0, p_2, a_2), (\circ, c_1, \bar{o}_1) \rangle$
2	$\langle (u_3, d_1, p_4, a_2), (-, c_2, \bar{o}_2) \rangle$
3	$\langle (u_2, d_2, p_2, a_0), (+, c_3, \bar{o}_3) \rangle$

1	$\langle (u_1, d_0, p_2, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle \rangle$
2a	$\langle (u_4, d_0, p_0, a_0),\ \langle (dr, \text{true}, do).\rangle \rangle$
2b	$\langle (u_2, d_0, p_0, a_0),\ \langle (dr, \text{true}, d\bar{o}).\rangle \rangle$
2c	$\langle (u_0, d_2, p_0, a_0),\ \langle (dr, \text{true}, d\bar{o}).\rangle \rangle$
2d	$\langle (u_0, d_0, p_3, a_0),\ \langle (dr, \text{true}, d\bar{o}).\rangle \rangle$
2e	$\langle (u_0, d_0, p_1, a_0),\ \langle (dr, \text{true}, d\bar{o}).\rangle \rangle$
2f	$\langle (u_0, d_0, p_0, a_1),\ \langle (dr, \text{true}, d\bar{o}).\rangle \rangle$
3	$\langle (u_2, d_2, p_2, a_0),\ \langle (+, c_3, \bar{o}_3).\rangle \rangle$
2'	$\langle (u_0, d_0, p_0, a_0),\ \langle (-, c_2, \bar{o}_2).\rangle \rangle$

2) Obligation rules



1	$\langle(u_1, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle(u_4, d_0,$ p0$, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2b	$\langle(u_2, d_0, p_0, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2c	$\langle(u_0, d_2, p_0, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2d	$\langle(u_0, d_0, p_3, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2e	$\langle(u_0, d_0, p_1, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2f	$\langle(u_0, d_0, p_0, a_1),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
3	$\langle(u_2, d_2, p_2, a_0),\ \langle(+, c_3, \bar{o}_3).\rangle\rangle$
2'	$\langle(u_0, d_0, p_0, a_0),\ \langle(-, c_2, \bar{o}_2).\rangle\rangle$

We have to distinguish among four cases:

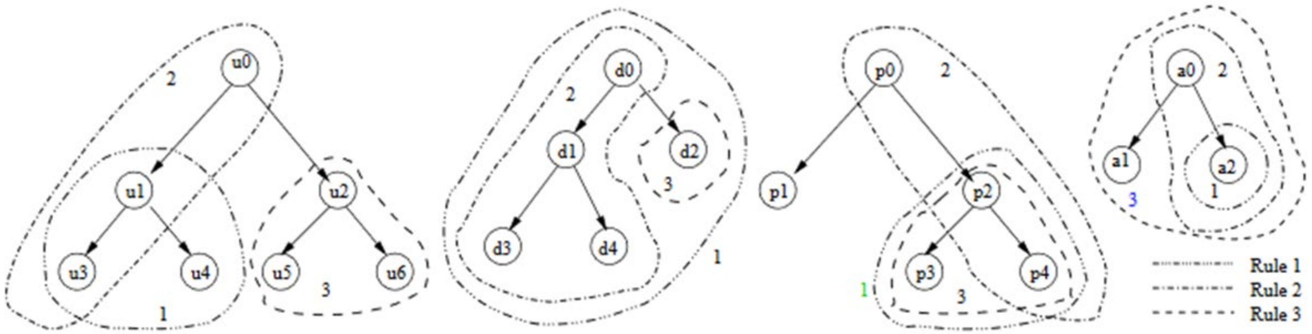
1. If there is no overlap with the next lower rule, we swap both rules.
2. If the scope of the floating rule is contained in the scope of the next rule, the qualifier of that rule is appended to the floating rule's qualifier and the obligation rule has reached its final position.
3. If the scope of the next rule is contained in the scope of the floating rule, we swap both rules but additionally append the qualifier of the floating rule to the qualifier sequence of the current rule.
4. If both rules overlap only partially, we swap the rules and additionally insert a new rule that deals with the overlap as follows:

$$overlap(\langle(u, d, p, a)\ seq_1\rangle, \langle(u', d', p', a')\ seq_2\rangle) =: \langle(u^*, d^*, p^*, a^*)\ seq_1\rangle$$

where $u^* = \begin{cases} u & \text{if } u \leq_U u' \\ u' & \text{otherwise} \end{cases}$,

and similarly for the other dimensions.

2) Obligation rules



We have to distinguish among four cases:

1. If there is no overlap with the next lower rule, we swap both rules.
2. If the scope of the floating rule is contained in the scope of the next rule, the qualifier of that rule is appended to the floating rule's qualifier and the obligation rule has reached its final position.
3. If the scope of the next rule is contained in the scope of the floating rule, we swap both rules but additionally append the qualifier of the floating rule to the qualifier sequence of the current rule.
4. If both rules overlap only partially, we swap the rules and additionally insert a new rule that deals with the overlap as follows:

$$overlap(\langle (u,d,p,a)\ seq_1\rangle, \langle (u',d',p',a')\ seq_2\rangle) =: \langle (u^*,d^*,p^*,a^*)\ seq_1\rangle$$

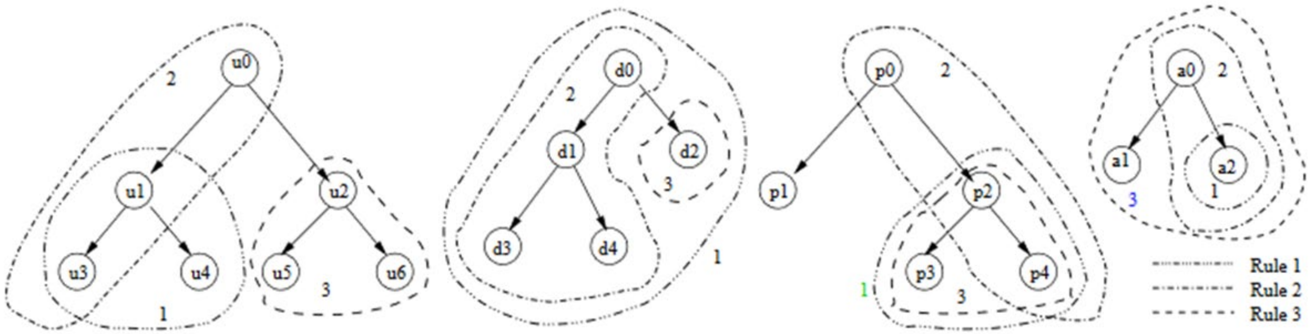
where $u^* = \begin{cases} u & \text{if } u \leq_U u' \\ u' & \text{otherwise} \end{cases}$,

and similarly for the other dimensions.

1	$\langle (u_1,d_0,p_2,a_2),\ \langle (\circ,c_1,\bar{o}_1).\rangle\rangle$
2a	$\langle (u_4,d_0,p_0,a_0),\ \langle (dr,\mathsf{true},\bar{do}).\rangle\rangle$
2b	$\langle (u_2,d_0,p_0,a_0),\ \langle (dr,\mathsf{true},\bar{do}).\rangle\rangle$
2c	$\langle (u_0,d_2,p_0,a_0),\ \langle (dr,\mathsf{true},\bar{do}).\rangle\rangle$
2d	$\langle (u_0,d_0,p_3,a_0),\ \langle (dr,\mathsf{true},\bar{do}).\rangle\rangle$
2e	$\langle (u_0,d_0,p_1,a_0),\ \langle (dr,\mathsf{true},\bar{do}).\rangle\rangle$
2f	$\langle (u_0,d_0,p_0,a_1),\ \langle (dr,\mathsf{true},\bar{do}).\rangle\rangle$
3	$\langle (u_2,d_2,p_2,a_0),\ \langle (+,c_3,\bar{o}_3).\rangle\rangle$
2'	$\langle (u_0,d_0,p_0,a_0),\ \langle (-,c_2,\bar{o}_2).\rangle\rangle$

2a'	$\langle (u_4,d_0,p_2,a_2),\ \langle (\circ,c_1,\bar{o}_1).\rangle\rangle$
2a	$\langle (u_4,d_0,p_0,a_0),\ \langle (dr,\mathsf{true},\bar{do}).\rangle\rangle$
1	$\langle (u_1,d_0,p_2,a_2),\ \langle (\circ,c_1,\bar{o}_1).\rangle\rangle$
2b	$\langle (u_2,d_0,p_0,a_0),\ \langle (dr,\mathsf{true},\bar{do}).\rangle\rangle$
	...

2) Obligation rules



We have to distinguish among four cases:

1. If there is no overlap with the next lower rule, we swap both rules.
2. If the scope of the floating rule is contained in the scope of the next rule, the qualifier of that rule is appended to the floating rule's qualifier and the obligation rule has reached its final position.
3. If the scope of the next rule is contained in the scope of the floating rule, we swap both rules but additionally append the qualifier of the floating rule to the qualifier sequence of the current rule.
4. If both rules overlap only partially, we swap the rules and additionally insert a new rule that deals with the overlap as follows:

$$overlap(\langle(u,d,p,a)\ seq_1\rangle, \langle(u',d',p',a')\ seq_2\rangle) =: \langle(u^*,d^*,p^*,a^*)\ seq_1\rangle$$

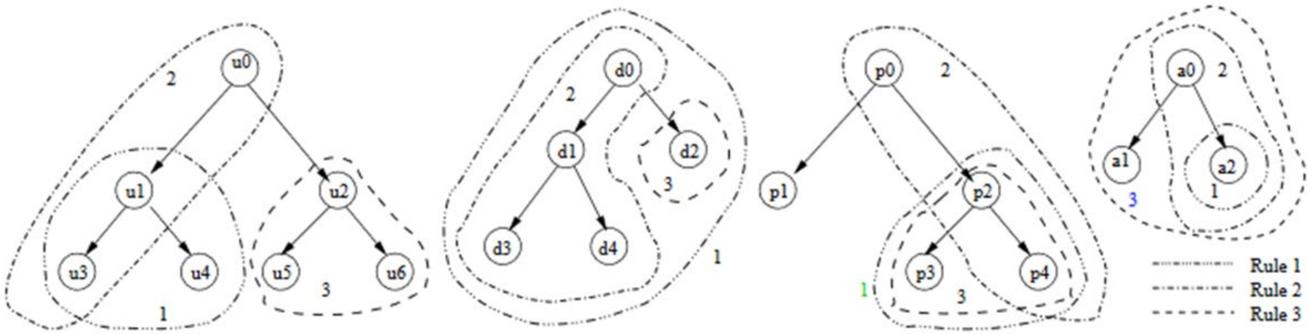
where $u^* = \begin{cases} u & \text{if } u \leq_U u' \\ u' & \text{otherwise} \end{cases}$,

and similarly for the other dimensions.

2a'	$\langle(u_4, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle(u_4, d_0, p0, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
1	$\boxed{\langle(u_1, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle}$
2b	$\boxed{\langle(u_2, d_0, p_0, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle}$
...	

2a'	$\langle(u_4, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle(u_4, d_0, p0, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2b	$\boxed{\langle(u_2, d_0, p_0, a_0),\ \langle(dr, \text{true}, do).\rangle\rangle}$
1	$\boxed{\langle(u_1, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle}$
2c	$\langle(u_0, d_2, p_0, a_0),\ \langle(dr, \text{true}, do).\rangle\rangle$
...	

2) Obligation rules



We have to distinguish among four cases:

1. If there is no overlap with the next lower rule, we swap both rules.
2. If the scope of the floating rule is contained in the scope of the next rule, the qualifier of that rule is appended to the floating rule's qualifier and the obligation rule has reached its final position.
3. If the scope of the next rule is contained in the scope of the floating rule, we swap both rules but additionally append the qualifier of the floating rule to the qualifier sequence of the current rule.
4. If both rules overlap only partially, we swap the rules and additionally insert a new rule that deals with the overlap as follows:

$$overlap(\langle(u,d,p,a)\ seq_1\rangle, \langle(u',d',p',a')\ seq_2\rangle) =: \langle(u^*,d^*,p^*,a^*)\ seq_1\rangle$$

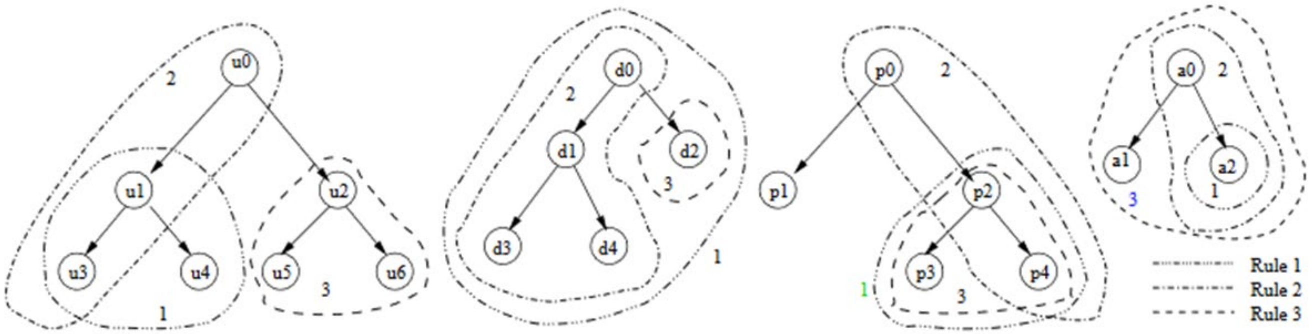
where $u^* = \begin{cases} u & \text{if } u \leq_U u' \\ u' & \text{otherwise} \end{cases}$

and similarly for the other dimensions.

2a'	$\langle(u_4,d_0,p_2,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
2a	$\langle(u_4,d_0,p_0,a_0),\ \langle(dr,\mathsf{true},\overline{do}).\rangle\rangle$
2b	$\langle(u_2,d_0,p_0,a_0),\ \langle(dr,\mathsf{true},\overline{do}).\rangle\rangle$
1	$\langle(u_1,d_0,p_2,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
2c	$\langle(u_0,d_2,p_0,a_0),\ \langle(dr,\mathsf{true},\overline{do}).\rangle\rangle$
...	

2a'	$\langle(u_4,d_0,p_2,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
2a	$\langle(u_4,d_0,p_0,a_0),\ \langle(dr,\mathsf{true},\overline{do}).\rangle\rangle$
2b	$\langle(u_2,d_0,p_0,a_0),\ \langle(dr,\mathsf{true},\overline{do}).\rangle\rangle$
2c'	$\langle(u_1,d_2,p_2,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
2c	$\langle(u_0,d_2,p_0,a_0),\ \langle(dr,\mathsf{true},do).\rangle\rangle$
1	$\langle(u_1,d_0,p_2,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
2d	$\langle(u_0,d_0,p_3,a_0),\ \langle(dr,\mathsf{true},do).\rangle\rangle$
...	

2) Obligation rules



We have to distinguish among four cases:

1. If there is no overlap with the next lower rule, we swap both rules.
2. If the scope of the floating rule is contained in the scope of the next rule, the qualifier of that rule is appended to the floating rule's qualifier and the obligation rule has reached its final position.
3. If the scope of the next rule is contained in the scope of the floating rule, we swap both rules but additionally append the qualifier of the floating rule to the qualifier sequence of the current rule.
4. If both rules overlap only partially, we swap the rules and additionally insert a new rule that deals with the overlap as follows:

$$overlap(\langle(u,d,p,a)\ seq_1\rangle, \langle(u',d',p',a')\ seq_2\rangle) =: \langle(u^*,d^*,p^*,a^*)\ seq_1\rangle$$

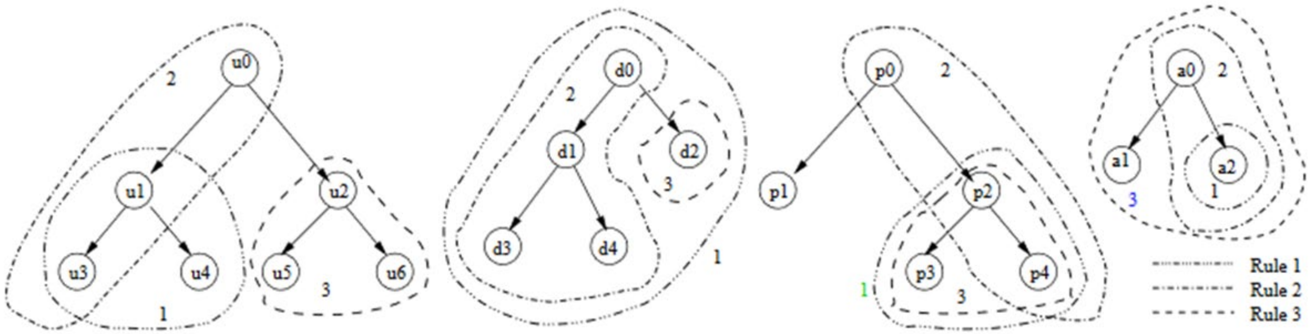
where $u^* = \begin{cases} u & \text{if } u \leq_U u' \\ u' & \text{otherwise} \end{cases}$,

and similarly for the other dimensions.

2a'	$\langle(u_4, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle(u_4, d_0,$ p0 $, a_0),\ \langle(dr, \text{true}, \overline{do}).\rangle\rangle$
2b	$\langle(u_2, d_0, p_0, a_0),\ \langle(dr, \text{true}, \overline{do}).\rangle\rangle$
2c'	$\langle(u_1, d_2, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2c	$\langle(u_0, d_2, p_0, a_0),\ \langle(dr, \text{true}, \overline{do}).\rangle\rangle$
1	$\langle(u_1, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2d	$\langle(u_0, d_0, p_3, a_0),\ \langle(dr, \text{true}, \overline{do}).\rangle\rangle$
	...

2a'	$\langle(u_4, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle(u_4, d_0,$ p0 $, a_0),\ \langle(dr, \text{true}, \overline{do}).\rangle\rangle$
2b	$\langle(u_2, d_0, p_0, a_0),\ \langle(dr, \text{true}, \overline{do}).\rangle\rangle$
2c'	$\langle(u_1, d_2, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2c	$\langle(u_0, d_2, p_0, a_0),\ \langle(dr, \text{true}, \overline{do}).\rangle\rangle$
2d'	$\langle(u_1, d_0, p_3, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2d	$\langle(u_0, d_0, p_3, a_0),\ \langle(dr, \text{true}, \overline{do}).\rangle\rangle$
1	$\langle(u_1, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2e	$\langle(u_0, d_0, p_1, a_0),\ \langle(dr, \text{true}, \overline{do}).\rangle\rangle$
2f	$\langle(u_0, d_0, p_0, a_1),\ \langle(dr, \text{true}, \overline{do}).\rangle\rangle$
	...

2) Obligation rules



We have to distinguish among four cases:

1. If there is no overlap with the next lower rule, we swap both rules.
2. If the scope of the floating rule is contained in the scope of the next rule, the qualifier of that rule is appended to the floating rule's qualifier and the obligation rule has reached its final position.
3. If the scope of the next rule is contained in the scope of the floating rule, we swap both rules but additionally append the qualifier of the floating rule to the qualifier sequence of the current rule.
4. If both rules overlap only partially, we swap the rules and additionally insert a new rule that deals with the overlap as follows:

$$overlap(\langle(u,d,p,a)\ seq_1\rangle, \langle(u',d',p',a')\ seq_2\rangle) =: \langle(u^*,d^*,p^*,a^*)\ seq_1\rangle$$

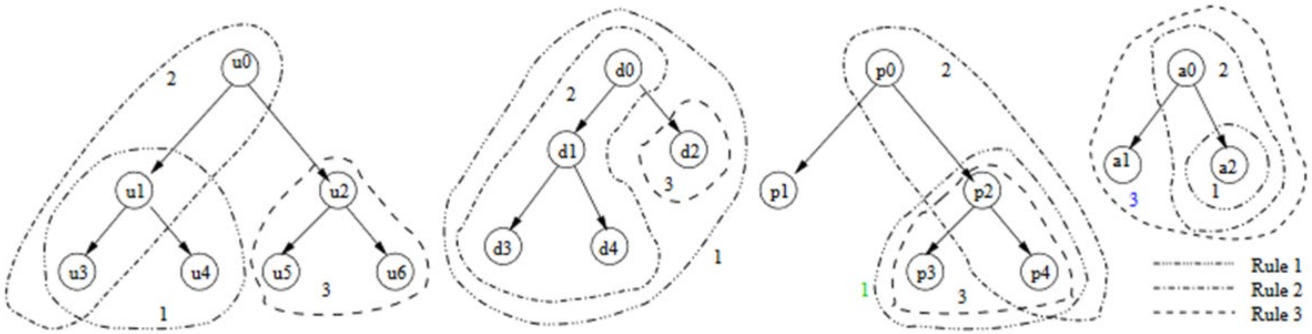
where $u^* = \begin{cases} u & \text{if } u \leq_U u' \\ u' & \text{otherwise} \end{cases}$,

and similarly for the other dimensions.

2a'	$\langle(u_4,d_0,p_2,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
2a	$\langle(u_4,d_0,p_0,a_0),\ \langle(dr,\text{true},\bar{do}).\rangle\rangle$
2b	$\langle(u_2,d_0,p_0,a_0),\ \langle(dr,\text{true},\bar{do}).\rangle\rangle$
2c'	$\langle(u_1,d_2,p_2,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
2c	$\langle(u_0,d_2,p_0,a_0),\ \langle(dr,\text{true},\bar{do}).\rangle\rangle$
2d'	$\langle(u_1,d_0,p_3,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
2d	$\langle(u_0,d_0,p_3,a_0),\ \langle(dr,\text{true},\bar{do}).\rangle\rangle$
1	$\langle(u_1,d_0,p_2,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
2e	$\langle(u_0,d_0,p_1,a_0),\ \langle(dr,\text{true},\bar{do}).\rangle\rangle$
2f	$\langle(u_0,d_0,p_0,a_1),\ \langle(dr,\text{true},\bar{do}).\rangle\rangle$
...	

2a'	$\langle(u_4,d_0,p_2,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
2a	$\langle(u_4,d_0,p_0,a_0),\ \langle(dr,\text{true},\bar{do}).\rangle\rangle$
2b	$\langle(u_2,d_0,p_0,a_0),\ \langle(dr,\text{true},\bar{do}).\rangle\rangle$
2c'	$\langle(u_1,d_2,p_2,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
2c	$\langle(u_0,d_2,p_0,a_0),\ \langle(dr,\text{true},\bar{do}).\rangle\rangle$
2d'	$\langle(u_1,d_0,p_3,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
2d	$\langle(u_0,d_0,p_3,a_0),\ \langle(dr,\text{true},\bar{do}).\rangle\rangle$
2e	$\langle(u_0,d_0,p_1,a_0),\ \langle(dr,\text{true},\bar{do}).\rangle\rangle$
2f	$\langle(u_0,d_0,p_0,a_1),\ \langle(dr,\text{true},\bar{do}).\rangle\rangle$
1	$\langle(u_1,d_0,p_2,a_2),\ \langle(\circ,c_1,\bar{o}_1).\rangle\rangle$
3	$\langle(u_2,d_2,p_2,a_0),\ \langle(+,c_3,\bar{o}_3).\rangle\rangle$
...	

2) Obligation rules



We have to distinguish among four cases:

1. If there is no overlap with the next lower rule, we swap both rules.
2. If the scope of the floating rule is contained in the scope of the next rule, the qualifier of that rule is appended to the floating rule's qualifier and the obligation rule has reached its final position.
3. If the scope of the next rule is contained in the scope of the floating rule, we swap both rules but additionally append the qualifier of the floating rule to the qualifier sequence of the current rule.
4. If both rules overlap only partially, we swap the rules and additionally insert a new rule that deals with the overlap as follows:

$$overlap(\langle (u,d,p,a)\ seq_1\rangle, \langle (u',d',p',a')\ seq_2\rangle) =: \langle (u^*,d^*,p^*,a^*)\ seq_1\rangle$$

where $u^* = \begin{cases} u & \text{if } u \leq_U u' \\ u' & \text{otherwise} \end{cases}$

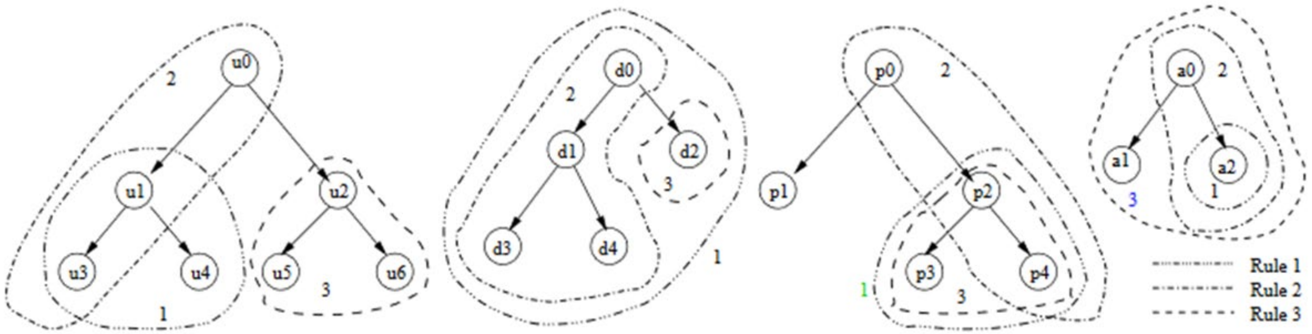
and similarly for the other dimensions.

2a'	$\langle (u_4, d_0, p_2, a_2), \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle (u_4, d_0, p0, a_0), \langle (dr, \text{true}, \bar{d}o).\rangle\rangle$
2b	$\langle (u_2, d_0, p_0, a_0), \langle (dr, \text{true}, \bar{d}o).\rangle\rangle$
2c'	$\langle (u_1, d_2, p_2, a_2), \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2c	$\langle (u_0, d_2, p_0, a_0), \langle (dr, \text{true}, \bar{d}o).\rangle\rangle$
2d'	$\langle (u_1, d_0, p_3, a_2), \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2d	$\langle (u_0, d_0, p_3, a_0), \langle (dr, \text{true}, \bar{d}o).\rangle\rangle$
2e	$\langle (u_0, d_0, p_1, a_0), \langle (dr, \text{true}, \bar{d}o).\rangle\rangle$
2f	$\langle (u_0, d_0, p_0, a_1), \langle (dr, \text{true}, \bar{d}o).\rangle\rangle$
1	$\langle (u_1, d_0, p_2, a_2), \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
3	$\langle (u_2, d_2, p_2, a_0), \langle (+, c_3, \bar{o}_3).\rangle\rangle$

...

2a'	$\langle (u_4, d_0, p_2, a_2), \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle (u_4, d_0, p0, a_0), \langle (dr, \text{true}, \bar{d}o).\rangle\rangle$
2b	$\langle (u_2, d_0, p_0, a_0), \langle (dr, \text{true}, \bar{d}o).\rangle\rangle$
2c'	$\langle (u_1, d_2, p_2, a_2), \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2c	$\langle (u_0, d_2, p_0, a_0), \langle (dr, \text{true}, \bar{d}o).\rangle\rangle$
2d'	$\langle (u_1, d_0, p_3, a_2), \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2d	$\langle (u_0, d_0, p_3, a_0), \langle (dr, \text{true}, \bar{d}o).\rangle\rangle$
2e	$\langle (u_0, d_0, p_1, a_0), \langle (dr, \text{true}, \bar{d}o).\rangle\rangle$
2f	$\langle (u_0, d_0, p_0, a_1), \langle (dr, \text{true}, \bar{d}o).\rangle\rangle$
3	$\langle (u_2, d_2, p_2, a_0), \langle (+, c_3, \bar{o}_3).\rangle\rangle$
1	$\langle (u_1, d_0, p_2, a_2), \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2'	$\langle (u_0, d_0, p_0, a_0), \langle (-, c_2, \bar{o}_2).\rangle\rangle$

2) Obligation rules



We have to distinguish among four cases:

1. If there is no overlap with the next lower rule, we swap both rules.
2. If the scope of the floating rule is contained in the scope of the next rule, the qualifier of that rule is appended to the floating rule's qualifier and the obligation rule has reached its final position.
3. If the scope of the next rule is contained in the scope of the floating rule, we swap both rules but additionally append the qualifier of the floating rule to the qualifier sequence of the current rule.
4. If both rules overlap only partially, we swap the rules and additionally insert a new rule that deals with the overlap as follows:

$$overlap(\langle (u,d,p,a)\ seq_1\rangle, \langle (u',d',p',a')\ seq_2\rangle) =: \langle (u^*,d^*,p^*,a^*)\ seq_1\rangle$$

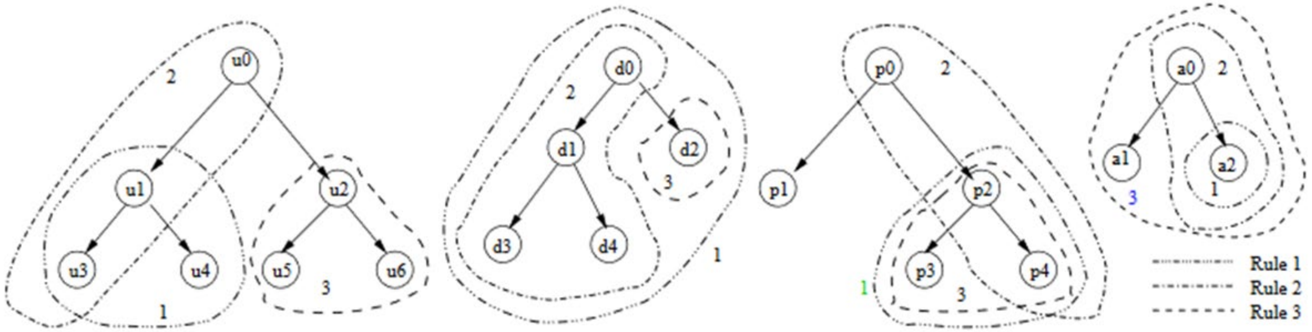
where $u^* = \begin{cases} u & \text{if } u \leq_U u' \\ u' & \text{otherwise} \end{cases}$,

and similarly for the other dimensions.

2a'	$\langle (u_4, d_0, p_2, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle (u_4, d_0,$ **p0**$, a_0),\ \langle (dr, \text{true}, \bar{do}).\rangle\rangle$
2b	$\langle (u_2, d_0, p_0, a_0),\ \langle (dr, \text{true}, \bar{do}).\rangle\rangle$
2c'	$\langle (u_1, d_2, p_2, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2c	$\langle (u_0, d_2, p_0, a_0),\ \langle (dr, \text{true}, \bar{do}).\rangle\rangle$
2d'	$\langle (u_1, d_0, p_3, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2d	$\langle (u_0, d_0, p_3, a_0),\ \langle (dr, \text{true}, \bar{do}).\rangle\rangle$
2e	$\langle (u_0, d_0, p_1, a_0),\ \langle (dr, \text{true}, \bar{do}).\rangle\rangle$
2f	$\langle (u_0, d_0, p_0, a_1),\ \langle (dr, \text{true}, \bar{do}).\rangle\rangle$
3	$\langle (u_2, d_2, p_2, a_0),\ \langle (+, c_3, \bar{o}_3).\rangle\rangle$
1	$\langle (u_1, d_0, p_2, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2'	$\langle (u_0, d_0, p_0, a_0),\ \langle (-, c_2, \bar{o}_2).\rangle\rangle$

2a'	$\langle (u_4, d_0, p_2, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle (u_4, d_0,$ **p0**$, a_0),\ \langle (dr, \text{true}, \bar{do}).\rangle\rangle$
2b	$\langle (u_2, d_0, p_0, a_0),\ \langle (dr, \text{true}, \bar{do}).\rangle\rangle$
2c'	$\langle (u_1, d_2, p_2, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2c	$\langle (u_0, d_2, p_0, a_0),\ \langle (dr, \text{true}, \bar{do}).\rangle\rangle$
2d'	$\langle (u_1, d_0, p_3, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2d	$\langle (u_0, d_0, p_3, a_0),\ \langle (dr, \text{true}, \bar{do}).\rangle\rangle$
2e	$\langle (u_0, d_0, p_1, a_0),\ \langle (dr, \text{true}, \bar{do}).\rangle\rangle$
2f	$\langle (u_0, d_0, p_0, a_1),\ \langle (dr, \text{true}, \bar{do}).\rangle\rangle$
3	$\langle (u_2, d_2, p_2, a_0),\ \langle (+, c_3, \bar{o}_3).\rangle\rangle$
1'	$\langle (u_1, d_0, p_2, a_2)\ \langle (\circ, c_1, \bar{o}_1); (-, c_2, \bar{o}_2).\rangle\rangle$
2'	$\langle (u_0, d_0, p_0, a_0),\ \langle (-, c_2, \bar{o}_2).\rangle\rangle$

3) Allow rules



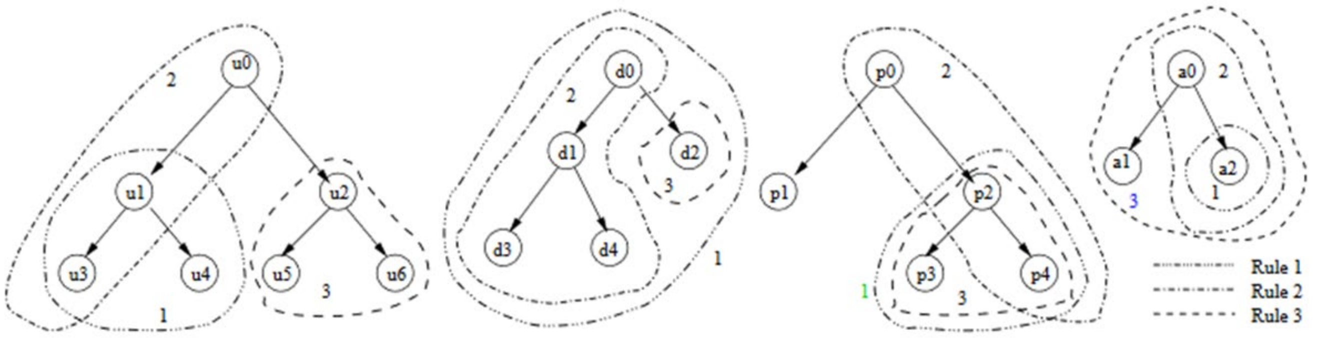
1. Float up allow rules until a rule is reached whose scope comprises the scope of the allow rule

2a'	$\langle (u_4, d_0, p_2, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle (u_4, d_0, \mathbf{p0}, a_0),\ \langle (dr, \mathsf{true}, \bar{do}).\rangle\rangle$
2b	$\langle (u_2, d_0, p_0, a_0),\ \langle (dr, \mathsf{true}, \bar{do}).\rangle\rangle$
2c'	$\langle (u_1, d_2, p_2, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2c	$\langle (u_0, d_2, p_0, a_0),\ \langle (dr, \mathsf{true}, \bar{do}).\rangle\rangle$
2d'	$\langle (u_1, d_0, p_3, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2d	$\langle (u_0, d_0, p_3, a_0),\ \langle (dr, \mathsf{true}, \bar{do}).\rangle\rangle$
2e	$\langle (u_0, d_0, p_1, a_0),\ \langle (dr, \mathsf{true}, \bar{do}).\rangle\rangle$
2f	$\langle (u_0, d_0, p_0, a_1),\ \langle (dr, \mathsf{true}, \bar{do}).\rangle\rangle$
3	$\langle (u_2, d_2, p_2, a_0),\ \langle (+, c_3, \bar{o}_3).\rangle\rangle$
1'	$\langle (u_1, d_0, p_2, a_2)\ \langle (\circ, c_1, o_1); (-, c_2, o_2).\rangle\rangle$
2'	$\langle (u_0, d_0, p_0, a_0),\ \langle (-, c_2, \bar{o}_2).\rangle\rangle$

Note: If the step is applied to the letter, then rule 3 should be below rule 2c. But… (see next slide)

3	$\langle (u_2, d_2, p_2, a_0),\ \langle (+, c_3, \bar{o}_3).\rangle\rangle$
2a'	$\langle (u_4, d_0, p_2, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle (u_4, d_0, \mathbf{p0}, a_0),\ \langle (dr, \mathsf{true}, \bar{do}).\rangle\rangle$
2b	$\langle (u_2, d_0, p_0, a_0),\ \langle (dr, \mathsf{true}, \bar{do}).\rangle\rangle$
2c'	$\langle (u_1, d_2, p_2, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2c	$\langle (u_0, d_2, p_0, a_0),\ \langle (dr, \mathsf{true}, \bar{do}).\rangle\rangle$
2d'	$\langle (u_1, d_0, p_3, a_2),\ \langle (\circ, c_1, \bar{o}_1).\rangle\rangle$
2d	$\langle (u_0, d_0, p_3, a_0),\ \langle (dr, \mathsf{true}, \bar{do}).\rangle\rangle$
2e	$\langle (u_0, d_0, p_1, a_0),\ \langle (dr, \mathsf{true}, \bar{do}).\rangle\rangle$
2f	$\langle (u_0, d_0, p_0, a_1),\ \langle (dr, \mathsf{true}, \bar{do}).\rangle\rangle$
1'	$\langle (u_1, d_0, p_2, a_2),\ \langle (\circ, c_1, o_1); (-, c_2, o_2).\rangle\rangle$
2'	$\langle (u_0, d_0, p_0, a_0),\ \langle (-, c_2, \bar{o}_2).\rangle\rangle$

But…



- The obtained rule set should be equivalent to the original one
- Consider request <u2,d2,p2,a0>

1	$\langle(u_1, d_0, p_2, a_2), (\circ, c_1, \bar{o}_1)\rangle$
2	$\langle(u_3, d_1, p_4, a_2), (-, c_2, \bar{o}_2)\rangle$
3	$\langle(u_2, d_2, p_2, a_0), (+, c_3, \bar{o}_3)\rangle$

(+,o3)

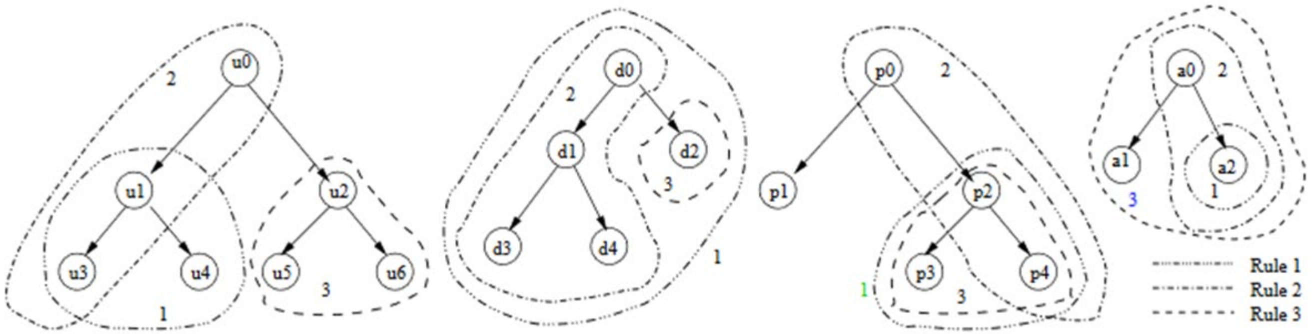
2a'	$\langle(u_4, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle(u_4, d_0,$ **p0**$, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2b	$\langle(u_2, d_0, p_0, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2c'	$\langle(u_1, d_2, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2c	$\langle(u_0, d_2, p_0, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
3	$\langle(u_2, d_2, p_2, a_0),\ \langle(+, c_3, \bar{o}_3).\rangle\rangle$
2d'	$\langle(u_1, d_0, p_3, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2d	$\langle(u_0, d_0, p_3, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2e	$\langle(u_0, d_0, p_1, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2f	$\langle(u_0, d_0, p_0, a_1),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
1'	$\langle(u_1, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1); (-, c_2, \bar{o}_2).\rangle\rangle$
2'	$\langle(u_0, d_0, p_0, a_0),\ \langle(-, c_2, \bar{o}_2).\rangle\rangle$

(dr,do)

3	$\langle(u_2, d_2, p_2, a_0),\ \langle(+, c_3, \bar{o}_3).\rangle\rangle$
2a'	$\langle(u_4, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2a	$\langle(u_4, d_0,$ **p0**$, a_0),\ \langle(dr, \text{true}, do).\rangle\rangle$
2b	$\langle(u_2, d_0, p_0, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2c'	$\langle(u_1, d_2, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2c	$\langle(u_0, d_2, p_0, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2d'	$\langle(u_1, d_0, p_3, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle\rangle$
2d	$\langle(u_0, d_0, p_3, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2e	$\langle(u_0, d_0, p_1, a_0),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
2f	$\langle(u_0, d_0, p_0, a_1),\ \langle(dr, \text{true}, \bar{do}).\rangle\rangle$
1'	$\langle(u_1, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1); (-, c_2, \bar{o}_2).\rangle\rangle$
2'	$\langle(u_0, d_0, p_0, a_0),\ \langle(-, c_2, \bar{o}_2).\rangle\rangle$

(+,o3)

3) Allow rules



1. Float up allow rules until a (obligation) rule is reached whose scope comprises the scope of the allow rule

2a'	$\langle(u_4, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle$
2a	$\langle(u_4, d_0, \mathbf{p0}, a_0),\ \langle(dr, \mathsf{true}, \bar{do}).\rangle$
2b	$\langle(u_2, d_0, p_0, a_0),\ \langle(dr, \mathsf{true}, \bar{do}).\rangle$
2c'	$\langle(u_1, d_2, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle$
2c	$\langle(u_0, d_2, p_0, a_0),\ \langle(dr, \mathsf{true}, \bar{do}).\rangle$
2d'	$\langle(u_1, d_0, p_3, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle$
2d	$\langle(u_0, d_0, p_3, a_0),\ \langle(dr, \mathsf{true}, \bar{do}).\rangle$
2e	$\langle(u_0, d_0, p_1, a_0),\ \langle(dr, \mathsf{true}, \bar{do}).\rangle$
2f	$\langle(u_0, d_0, p_0, a_1),\ \langle(dr, \mathsf{true}, \bar{do}).\rangle$
3	$\langle(u_2, d_2, p_2, a_0),\ \langle(+, c_3, \bar{o}_3).\rangle$
1'	$\langle(u_1, d_0, p_2, a_2)\ \langle(\circ, c_1, \bar{o}_1); (-, c_2, \bar{o}_2).\rangle$
2'	$\langle(u_0, d_0, p_0, a_0),\ \langle(-, c_2, \bar{o}_2).\rangle$

3	$\langle(u_2, d_2, p_2, a_0),\ \langle(+, c_3, \bar{o}_3).\rangle$
2a'	$\langle(u_4, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle$
2a	$\langle(u_4, d_0, \mathbf{p0}, a_0),\ \langle(dr, \mathsf{true}, \bar{do}).\rangle$
2b	$\langle(u_2, d_0, p_0, a_0),\ \langle(dr, \mathsf{true}, \bar{do}).\rangle$
2c'	$\langle(u_1, d_2, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle$
2c	$\langle(u_0, d_2, p_0, a_0),\ \langle(dr, \mathsf{true}, \bar{do}).\rangle$
2d'	$\langle(u_1, d_0, p_3, a_2),\ \langle(\circ, c_1, \bar{o}_1).\rangle$
2d	$\langle(u_0, d_0, p_3, a_0),\ \langle(dr, \mathsf{true}, \bar{do}).\rangle$
2e	$\langle(u_0, d_0, p_1, a_0),\ \langle(dr, \mathsf{true}, \bar{do}).\rangle$
2f	$\langle(u_0, d_0, p_0, a_1),\ \langle(dr, \mathsf{true}, \bar{do}).\rangle$
1'	$\langle(u_1, d_0, p_2, a_2),\ \langle(\circ, c_1, \bar{o}_1); (-, c_2, \bar{o}_2).\rangle$
2'	$\langle(u_0, d_0, p_0, a_0),\ \langle(-, c_2, \bar{o}_2).\rangle$

Comparison of Qualifier Sequences

Goal: given two sequences of qualifiers, check whether there is a refinement

For each qualifier in one sequence and in descending order,

- check the qualifiers in the second sequence for refinement
  - The conditions of the two qualifiers should be concurrently satisfied
- until a qualifier whose condition implies the qualifier's condition of the first sequence is reached

Comparison in action

Consider the following sequences of qualifiers.

$$(r_1, c_1 \wedge c_2 \wedge c_3, \bar{o}_1); (r_2, c_1 \wedge c_2, \bar{o}_2); (r_3, c_2, \bar{o}_3); (r_4, c_1, \bar{o}_4); (dr_1, \mathsf{true}, \bar{do}_1\rangle \tag{1}$$

$$(r'_1, c_1 \wedge c_3, \bar{o}'_1); (r'_2, c_3, \bar{o}'_2); (r'_3, c_1, \bar{o}'_3); (dr_2, \mathsf{true}, \bar{do}_2) \tag{2}$$

Does sequence (2) refine sequence (1)?

Comparison (1)

$$\boxed{(r_1, c_1 \wedge c_2 \wedge c_3, \bar{o}_1)}; (r_2, c_1 \wedge c_2, \bar{o}_2); (r_3, c_2, \bar{o}_3); (r_4, c_1, \bar{o}_4); (dr_1, \mathsf{true}, \bar{do}_1\rangle \tag{1}$$

$$\boxed{(r'_1, c_1 \wedge c_3, \bar{o}'_1)}; (r'_2, c_3, \bar{o}'_2); (r'_3, c_1, \bar{o}'_3); (dr_2, \mathsf{true}, \bar{do}_2) \tag{2}$$

- Take the first qualifier of sequence (2) and assume that condition $c_1 \wedge c_3$ is true.
- Condition $c_1 \wedge c_2 \wedge c_3$ of the first qualifier $(r_1, c_1 \wedge c_2 \wedge c_3, \bar{o}_1)$ may be true concurrently
- Compare (r1, o1) and (r'1, o'1):
  - if r1 =/= o (don't care) then r'1 = r1?
  - o'1 refines o1?
- If this holds, continue the comparison with the next qualifier in sequence (1).

Comparison (2)

$$\boxed{(r_1, c_1 \wedge c_2 \wedge c_3, \bar{o}_1)}; \boxed{(r_2, c_1 \wedge c_2, \bar{o}_2)}; \boxed{(r_3, c_2, \bar{o}_3)}; (r_4, c_1, \bar{o}_4); (dr_1, \text{true}, \bar{do}_1\rangle \tag{1}$$

$$\boxed{(r'_1, c_1 \wedge c_3, \bar{o}'_1)}; (r'_2, c_3, \bar{o}'_2); (r'_3, c_1, \bar{o}'_3); (dr_2, \text{true}, \bar{do}_2) \tag{2}$$

- Take $(r_2, c_1 \wedge c_2, \bar{o}_2)$ (and condition $c_1 \wedge c_3$ holds).
- At this point condition $((c_1 \wedge c_3) \wedge \neg(c_1 \wedge c_2 \wedge c_3))$ holds
- Qualifier $(r_2, c_1 \wedge c_2, \bar{o}_2)$ is applicable if $c_1 \wedge c_2$ is true
- Since $((c_1 \wedge c_3) \wedge \neg(c_1 \wedge c_2 \wedge c_3)) \Rightarrow \neg(c_1 \wedge c_2)$ then $c_1 \wedge c_2$ cannot be true
- Therefore, qualifier $(r_2, c_1 \wedge c_2, \bar{o}_2)$ is not applicable
- The same holds for qualifier $(r_3, c_2, \bar{o}_3)$

Comparison (3)

$$(r_1, c_1 \wedge c_2 \wedge c_3, \bar{o}_1); (r_2, c_1 \wedge c_2, \bar{o}_2); (r_3, c_2, \bar{o}_3); \boxed{(r_4, c_1, \bar{o}_4)}; (dr_1, \mathsf{true}, \bar{do}_1\rangle \quad (1)$$

$$\boxed{(r'_1, c_1 \wedge c_3, \bar{o}'_1)}; (r'_2, c_3, \bar{o}'_2); (r'_3, c_1, \bar{o}'_3); (dr_2, \mathsf{true}, \bar{do}_2) \quad (2)$$

- Take qualifier $(r_4, c_1, \bar{o}_4)$ (and condition $c_1 \wedge c_3$ holds).
- Since $c_1 \wedge c_3 \Rightarrow c_1$, qualifier $(r_4, c_1, \bar{o}_4)$ is applicable
- Compare (r4, o4) and (r’1, o’1):
  - if r4 =/= o (don’t care) then r’1 = r4?
  - o’1 refines o4?
- Because $c_1 \wedge c_3$ implies c_1, no remaining elements in sequence (1) must be checked

Comparison (4)

$$\boxed{(r_1, c_1 \wedge c_2 \wedge c_3, \bar{o}_1); (r_2, c_1 \wedge c_2, \bar{o}_2); (r_3, c_2, \bar{o}_3)}; (r_4, c_1, \bar{o}_4); (dr_1, \mathsf{true}, \bar{do}_1\rangle \quad (1)$$

$$(r'_1, c_1 \wedge c_3, \bar{o}'_1); \boxed{(r'_2, c_3, \bar{o}'_2)}; (r'_3, c_1, \bar{o}'_3); (dr_2, \mathsf{true}, \bar{do}_2) \quad (2)$$

- Take the second qualifier of sequence (2)
- At this point, c_3 and $\neg(c_1 \wedge c_3)$ hold
- Since $c_1 \wedge c_3$ does not hold, qualifier $(r_1, c_1 \wedge c_2 \wedge c_3, \bar{o}_1)$ is not applicable
- Since *NOT(c1 Λ c3) Λ c3 holds*, also qualifier $(r_2, c_1 \wedge c_2, \bar{o}_2)$ is not applicable
- Since *c2 can be true*, qualifier $(r_3, c_2, \bar{o}_3)$ is applicable
- Compare (r3, o3) and (r'2, o'2):
  - if r'2 =/= o (don't care) then r'2 = r3?
  - o'2 refines o3?
- If this holds, continue the comparison with the next qualifier in sequence (1).

Comparison (5)

$$(r_1, c_1 \wedge c_2 \wedge c_3, \bar{o}_1); (r_2, c_1 \wedge c_2, \bar{o}_2); (r_3, c_2, \bar{o}_3); \boxed{(r_4, c_1, \bar{o}_4); (dr_1, \text{true}, \bar{do}_1\rangle} \quad (1)$$

$$(r'_1, c_1 \wedge c_3, \bar{o}'_1); \boxed{(r'_2, c_3, \bar{o}'_2)}; (r'_3, c_1, \bar{o}'_3); (dr_2, \text{true}, \bar{do}_2) \quad (2)$$

- Take qualifier $(r_4, c_1, \bar{o}_4)$
- At this point, c3 Λ NOT(c1 Λ c3) Λ NOT(c2) holds
- Because of c1 cannot be true, qualifier $(r_4, c_1, \bar{o}_4)$ is not applicable
- Qualifier $(dr_1, \text{true}, do_1\rangle$ is applicable
- Compare (dr1, do1) and (r'2, o'2):
  - if r'2 =/= o (don't care) then r'2 = dr1?
  - o'2 refines do1?
- If this holds, continue the comparison with the next qualifier in sequence (2).

Comparison (6)

- Check the remaining elements in sequence (2)

Satisfied Condition	Result given by seq (1)	Result given by seq (2)
$c_1 \wedge c_2 \wedge c_3$	$(r_1, \bar{o}_1)$	$(r'_1, \bar{o}'_1)$
$c_1 \wedge c_3$	$(r_4, \bar{o}_4)$	$(r'_1, \bar{o}'_1)$
$c_2 \wedge c_3$	$(r_3, \bar{o}_3)$	$(r'_2, \bar{o}'_2)$
c_3	$(dr_1, \bar{do}_1)$	$(r'_2, \bar{o}'_2)$
$c_1 \wedge c_2$	$(r_2, \bar{o}_2)$	$(r'_3, \bar{o}'_3)$
c_1	$(r_4, \bar{o}_4)$	$(r'_3, \bar{o}'_3)$
c_2	$(r_3, \bar{o}_3)$	$(dr_2, \bar{do}_2)$
- -	$(dr_1, \bar{do}_1)$	$(dr_2, \bar{do}_2)$

$$(r_1, c_1 \wedge c_2 \wedge c_3, \bar{o}_1); (r_2, c_1 \wedge c_2, \bar{o}_2); (r_3, c_2, \bar{o}_3); (r_4, c_1, \bar{o}_4); (dr_1, \mathsf{true}, \bar{do}_1\rangle \quad (1)$$

$$(r'_1, c_1 \wedge c_3, \bar{o}'_1); (r'_2, c_3, \bar{o}'_2); (r'_3, c_1, \bar{o}'_3); (dr_2, \mathsf{true}, \bar{do}_2) \quad (2)$$

Comparison of extended rule lists

Goal: check for refinement of two privacy policies by comparing their normalized, scope-based rule lists.

- If there is refinement for the qualifier sequences of all “matching” rules, then there is policy refinement.

Comparison of extended rule lists

Let SR1 and SR2 denote two scope-based rule lists.

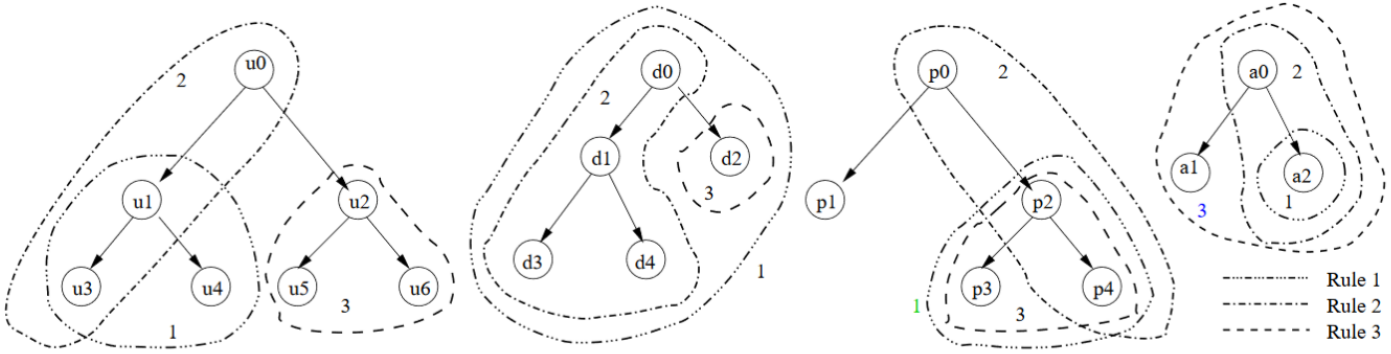
- For each rule σ2=〈(u2,d2,p2,a2),seq2〉 in SR2:
  - Process SR1 in descending order:
    - For each overlapping rule σ1=〈(u1,d1,p1,a1),seq1〉 in SR1:
      - Check whether the qualifier sequences seq2 and seq1 constitute a refinement
      - If there is no refinement, the algorithm stops and returns false.
      - If scope(σ2)⊆scope(σ1), the processing of rule σ2 terminates.

This processing always terminates because every SR ends with rule(s) covering the entire hierarchies (by construction)

Comparison of extended rule lists

1	$\langle(u_2, d_2, p_2, a_0),\ seq_1\rangle$			
2	$\langle(u_4, d_0, p_2, a_2),\ seq_2\rangle$			
3	$\langle(u_4, d_0, p_0, a_0),\ seq_3\rangle$			
4	$\langle(u_1, d_0, p_2, a_2),\ seq_4\rangle$		1’	$\langle(u_4, d_0, p_0, a_0),\ seq'_1\rangle$
5	$\langle(u_0, d_0, p_0, a_0),\ seq_5\rangle$		2’	$\langle(u_0, d_0, p_0, a_0),\ seq'_2\rangle$
	List SR_1			List SR_2

Comparison of extended rule lists



	List SR_1		List SR_2
1	$\langle (u_2, d_2, p_2, a_0),\ seq_1 \rangle$		
2	$\langle (u_4, d_0, p_2, a_2),\ seq_2 \rangle$		
3	$\langle (u_4, d_0, p_0, a_0),\ seq_3 \rangle$		
4	$\langle (u_1, d_0, p_2, a_2),\ seq_4 \rangle$	1'	$\langle (u_4, d_0, p_0, a_0),\ seq'_1 \rangle$
5	$\langle (u_0, d_0, p_0, a_0),\ seq_5 \rangle$	2'	$\langle (u_0, d_0, p_0, a_0),\ seq'_2 \rangle$

- Take rule 1'
- The first overlap is with rule 2 (scope(u4,d0,p2,a2)⊆scope(u4,d0,p0,a0))
- IF the qualifier sequences seq'1 and seq 2 are a refinement
  - THEN continue
  - ELSE stop (SR2 is not a refinement of SR1)
- The next overlap is with rule 3.
- After successful comparison we do not have to check the remaining rules in SR1 because the scope of rule 3 completely covers the scope of rule 1'.
- Continue with the second rule in SR2 and check overlap with the rules in SR1 in descending order.
- Because rule 2' has scope (u0,d0,p0,a0), the qualifier sequence of every rule in SR1 must be checked.