Testing Subcategory-Sensitive MatchSP Constraints

Nick Van Handel
August 2020

1. Subcategory-Sensitive MatchSP Constraints

This document describes the method used to test subcategory-sensitive MatchSP constraints in SPOT. The associated test file is *aug20_customMatchSP.html*. Labels for syntactic and prosodic trees correspond to the variable names used in the test file. Tableaux in this document are presented in the same order as tableaux in the test file. The test file's tableaux should have the same violations as the tableaux in this document if the constraints are working properly.

Custom Match constraints in SPOT include **relation-based settings** that allow the user to decide whether Match is sensitive to the Maximal and/or Minimal status of syntactic and prosodic projections. When these settings are used, the tree geometry affects how Match constraints are evaluated. These options are as follows:

- Syntactic nodes:
  - Maximal:
    - Yes (= Maximal only)
    - No (= Non-maximal only)
    - Any (= both visible)
  - Minimal:
    - Yes, No, Any
- Prosodic subcategories:
  - Maximal:
    - Yes, No, Any
  - Minimal:
    - Yes, No, Any
- 3*3*3*3 = 81 MatchSP constraints

Only MatchSP constraints were tested. The test shows that the MatchSP constraints are working, and it is assumed that the MatchPS constraints will also work as expected, because they use the same subcategories. Moreover, the MatchPS constraints were previously tested by Max Tarlov; see the test file *moreMatchOptions.html*.

The Lexical and Overtly Headed parameters were not tested. In other words, all XPs in the syntactic input were considered visible to the MatchSP constraints (modulo subcategory-sensitive settings).

2. Method

In this section, I describe the process by which all 81 subcategory-sensitive versions of MatchXP were tested in SPOT to ensure that violations are assigned correctly according to the maximal and minimal specifications.

First, a syntactic input was created that contains all four possible combinations of ±Max, ±Min, as shown in Figure 1.

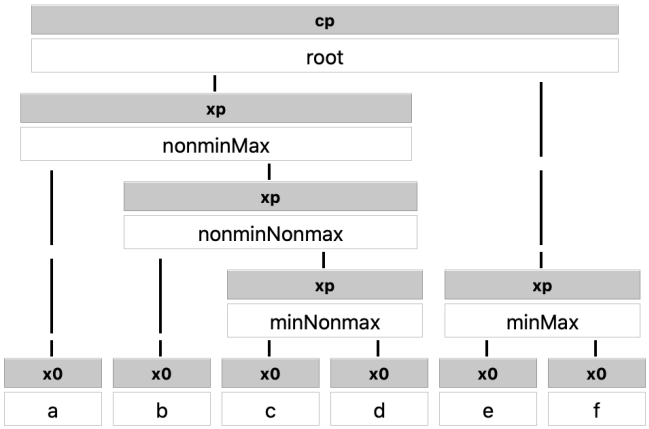

Figure 1. *stree1*, syntactic input with all combinations of ±Max, ±Min.

Next, two prosodic outputs were created. The first, *ptree1*, is a completely mismatched tree, shown in Figure 2. The second, *ptree2*, is a perfectly matched tree, shown in Figure 3.

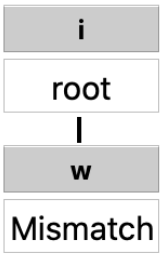

Figure 2. *ptree1*, a total mismatch.

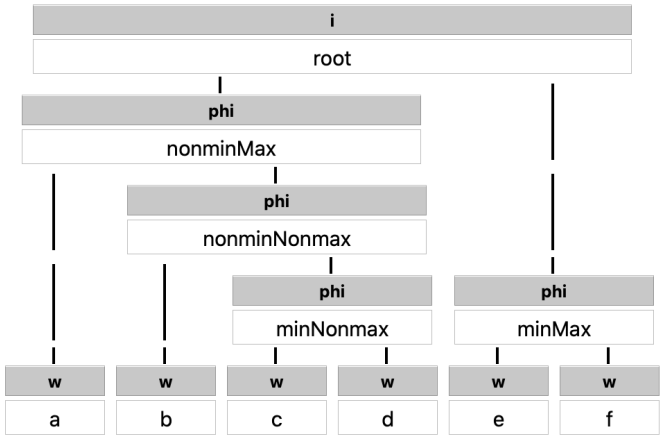

Figure 3. *ptree2*, a perfect match.

I then counted the number of violations that should be assigned by each Match constraint to these two candidates to make sure that they are working as expected. For instance, the general Match(XP,φ) should assign 4 violations to the mismatched candidate, because the input contains 4 XPs, each of which is not matched to a φ. However, Match($XP^{[+Max]}$,φ) should only assign 2 violations to the mismatch candidate, because there are only two maximal XPs that are not matched to a φ. Tableau 1 shows the expected violations for the general Match(XP,φ) and all combinations of ±Max, ±Min specified for XP, with no specifications for φ. Tableau 2 shows the same, but with all combinations of ±Max, ±Min for φ and no specifications for XP.

Tableau 1. Expected violations for MatchSP, all possible syntax specifications, no prosody specifications.

[a [b [c d]]] [e f] **all XPs**	MXP	MXP +MaxS	MXP -MaxS	MXP +MinS	MXP +MaxS +MinS	MXP -MaxS +MinS	MXP -MinS	MXP +MaxS -MinS	MXP -MaxS -MinS
mismatch	4	2	2	2	1	1	2	1	1
(a (b (c d))) (e f)	0	0	0	0	0	0	0	0	0

Tableau 2. Expected violations for MatchSP, all possible prosody specifications, no syntax specifications.

[a [b [c d]]] [e f] **all XPs**	MXP +MaxP	MXP -MaxP	MXP +MinP	MXP +MaxP +MinP	MXP -MaxP +MinP	MXP -MinP	MXP +MaxP -MinP	MXP -MaxP -MinP
mismatch	4	4	4	4	4	4	4	4
(a (b (c d))) (e f)	2	2	2	3	3	2	3	3

These tableaux were then compared to the first two tableaux in the test file *aug20_customMatchSP.html*. SPOT assigns the same violations, so the Match constraints are working properly.

2.1 Syntax: +Max, +Min

A similar process was carried out to test all the constraints in which subcategories are specified for *both* XP and φ. In each of the following subsections, a particular combination of subcategories of XPs was held constant and combined with each possible combination of subcategory settings for φ. In each case, four prosodic trees were created, such that the syntactic constituent targeted by Match was mapped to a φ that was [+Max, +Min], [+Max, -Min], [-Max, +Min], or [-Max, -Min]. This ensured that the test targeted all possible subcategory combinations.

In each subsection, the syntactic input is repeated, with boxes around those syntactic XPs targeted by the give subcategory specification. Additionally, each prosodic tree is shown, with a box around the φ corresponding to the syntactic XP targeted by Match. Then, a tableau with violations counted by hand is provided. Again, all tableaux were compared against the tableaux in the test file. The violation counts were identical, indicating that SPOT correctly assigns violations for all MatchSP constraints.

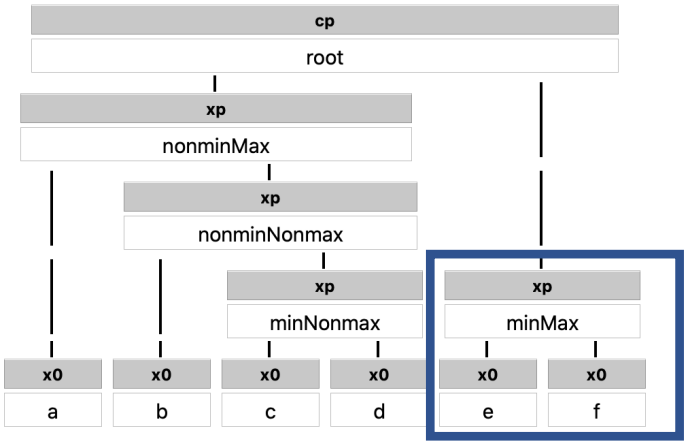

Figure 4. Syntactic input: [e f] is [+Max, +Min]

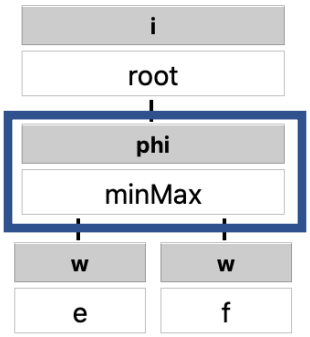

Figure 5. ptree3, (e f) is [+Max, +Min]

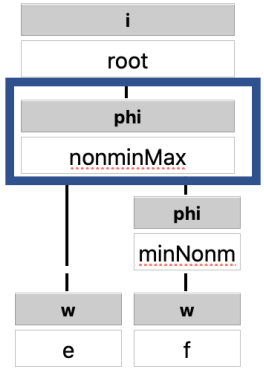

Figure 6. *ptree4*, (e f) is [+Max, -Min]

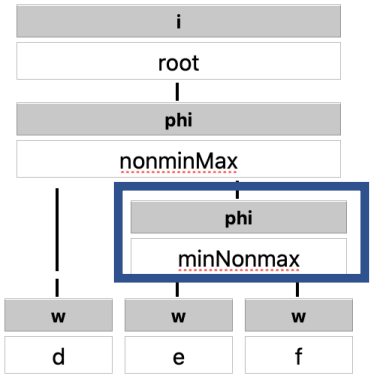

Figure 7. *ptree5*, (e f) is [-Max, +Min]

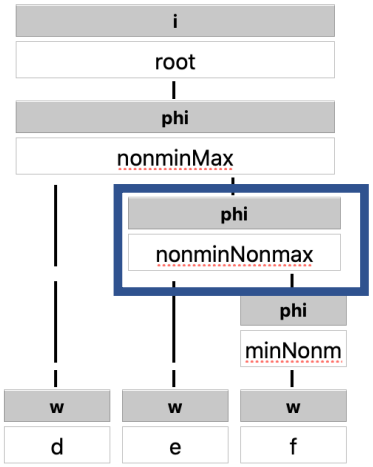

Figure 8. *ptree6*, (e f) is [-Max, -Min]

Tableau 3. Expected violations for MatchSP(XP$^{[+Max,+Min]}$), all possible prosody specifications

[a [b [c d]]] **[**e f**]** +MaxS +MinS	MXP +MaxS +MinS +MaxP	MXP +MaxS +MinS -MaxP	MXP +MaxS +MinS +MinP	MXP +MaxS +MinS +MaxP +MinP	MXP +MaxS +MinS -MaxP +MinP	MXP +MaxS +MinS -MinP	MXP +MaxS +MinS +MaxP -MinP	MXP +MaxS +MinS -MaxP -MinP
(d **(**e (f)**)**) **-MaxP** **-MinP**	1	0	1	1	1	0	1	0
(d **(**e f**)**) **-MaxP** **+MinP**	1	0	0	1	0	1	1	1
(e (f)**)** **+MaxP** **-MinP**	0	1	1	1	1	0	0	1
(e f**)** **+MaxP** **+MinP**	0	1	0	0	1	1	1	1
Mismatch	1	1	1	1	1	1	1	1

2.2 Syntax: +Max, -Min

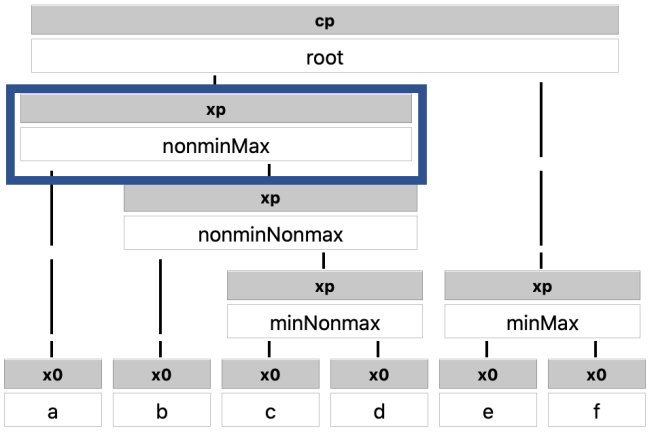

Figure 9. Syntactic input: [a b c d] is [+Max, -Min]

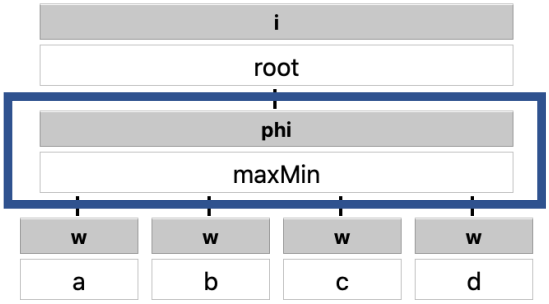

Figure 10. ptree7, (a b c d) is [+Max, +Min]

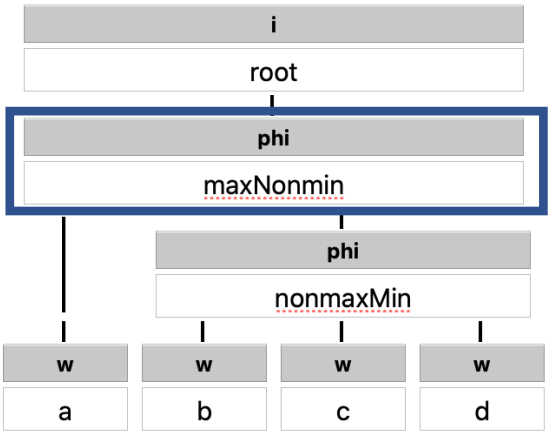

Figure 11. ptree8, (a b c d) is [+Max, -Min]

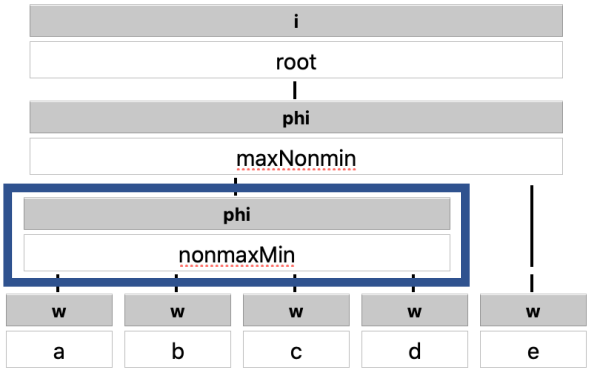

Figure 12. ptree9, (a b c d) is [-Max, +Min]

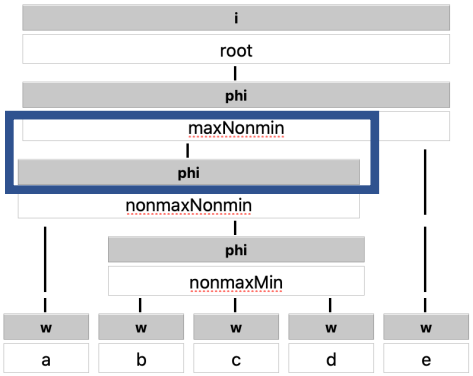

Figure 13. ptree10, (a b c d) is [-Max, -Min]

Tableau 4. Expected violations for MatchSP($XP^{[+Max,-Min]}$), all possible prosody specifications

[a [b [c d]]] [e f] +MaxS -MinS	MXP +MaxS -MinS +MaxP	MXP +MaxS -MinS -MaxP	MXP +MaxS -MinS +MinP	MXP +MaxS -MinS +MaxP +MinP	MXP +MaxS -MinS -MaxP +MinP	MXP +MaxS -MinS -MinP	MXP +MaxS -MinS +MaxP -MinP	MXP +MaxS -MinS -MaxP -MinP
((a (b c d)) e) **-MaxP** **-MinP**	1	0	1	1	1	0	1	0
((a b c d) e) **-MaxP** **+MinP**	1	0	0	1	0	1	1	1
(a (b c d)) **+MaxP** **-MinP**	0	1	1	1	1	0	0	1
(a b c d) **+MaxP** **+MinP**	0	1	0	0	1	1	1	1
Mismatch	1	1	1	1	1	1	1	1

2.3 Syntax: +Max, Any Min

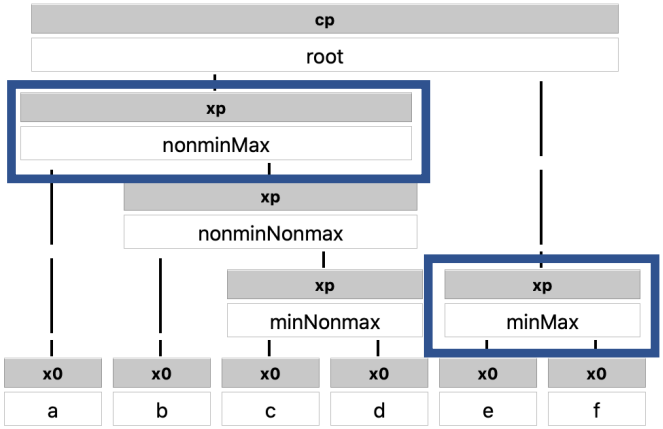

Figure 14. Syntactic input: [a b c d], [e f] are [+Max]

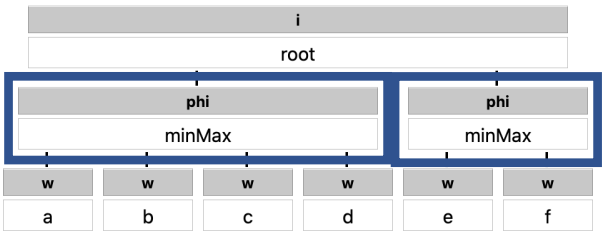

Figure 15. ptree11, (a b c d), (ef) are [+Max, +Min]

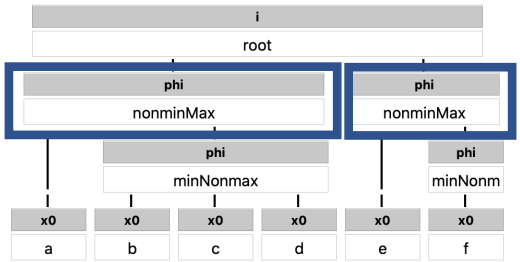

Figure 16. ptree12, (a b c d), (ef) are [+Max, -Min]

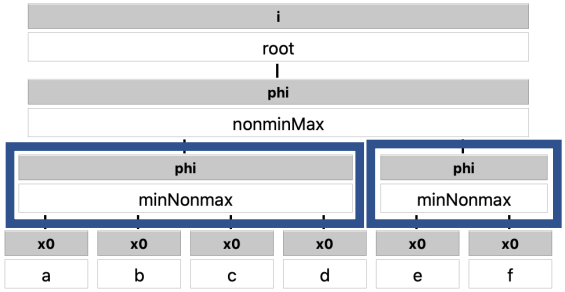

Figure 17. ptree13, (a b c d), (ef) are [-Max, +Min]

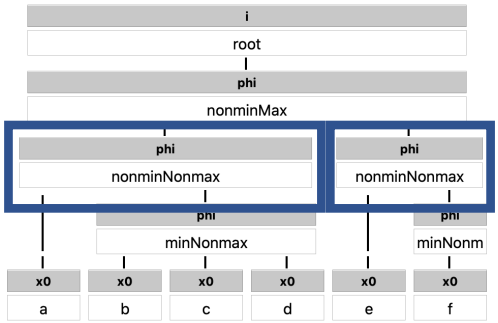

Figure 18. ptree14, (a b c d), (ef) are [-Max, -Min]

Tableau 5. Expected violations for MatchSP(XP[+Max]), all possible prosody specifications

[a [b [c d]]] [e f] +MaxS	MXP +MaxS +MaxP	MXP +MaxS -MaxP	MXP +MaxS +MinP	MXP +MaxS +MaxP +MinP	MXP +MaxS -MaxP +MinP	MXP +MaxS -MinP	MXP +MaxS +MaxP -MinP	MXP +MaxS -MaxP -MinP
((a (b c d)) (e (f))) **-MaxP -MinP**	2	0	2	2	2	0	2	0
((a b c d) (e f)) **-MaxP +MinP**	2	0	0	2	0	2	2	2
(a (b c d)) (e (f)) **+MaxP -MinP**	0	2	2	2	2	0	0	2
(a b c d) (e f) **+MaxP +MinP**	0	2	0	0	2	2	2	2
Mismatch	2	2	2	2	2	2	2	2

2.4 Syntax: -Max, +Min

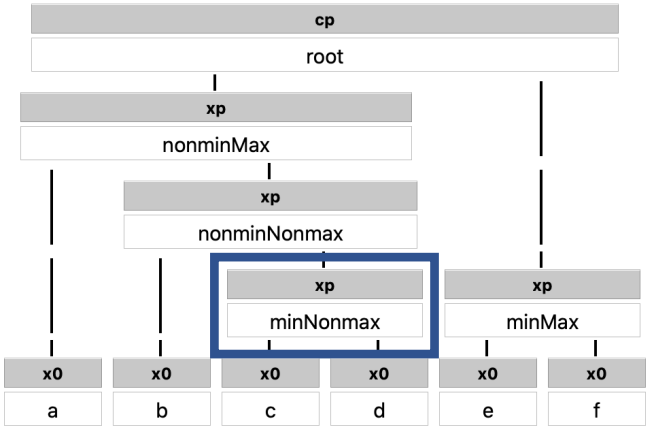

Figure 19. Syntactic input: [c d] is [-Max, +Min]

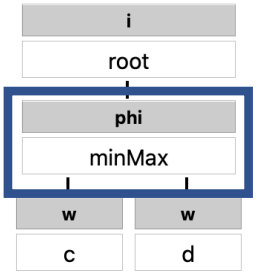

Figure 20. ptree15, (c d) is [+Max, +Min]

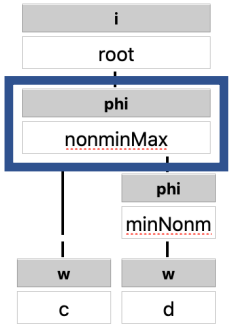

Figure 21. ptree16, (c d) is [+Max, -Min]

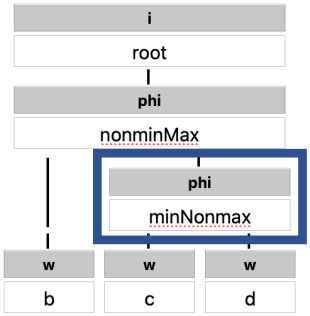

Figure 22. *ptree17*, (c d) is [-Max, +Min]

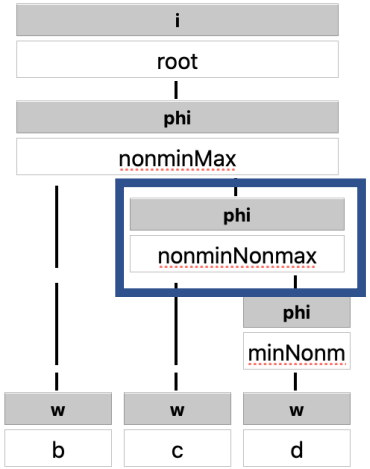

Figure 23. *ptree18*, (c d) is [-Max, -Min]

Tableau 6. Expected violations for MatchSP(XP[-Max,+Min]), all possible prosody specifications

[a [b [c d]]] [e f] -MaxS +MinS	MXP -MaxS +MinS +MaxP	MXP -MaxS +MinS -MaxP	MXP -MaxS +MinS +MinP	MXP -MaxS +MinS +MaxP +MinP	MXP -MaxS +MinS -MaxP +MinP	MXP -MaxS +MinS -MinP	MXP -MaxS +MinS +MaxP -MinP	MXP -MaxS +MinS -MaxP -MinP
(b (c (d))) **-MaxP** **-MinP**	1	0	1	1	1	0	1	0
(b (c d)) **-MaxP** **+MinP**	1	0	0	1	0	1	1	1
(c (d)) **+MaxP** **-MinP**	0	1	1	1	1	0	0	1
(c d) **+MaxP** **+MinP**	0	1	0	0	1	1	1	1
Mismatch	1	1	1	1	1	1	1	1

2.5 Syntax: -Max, -Min

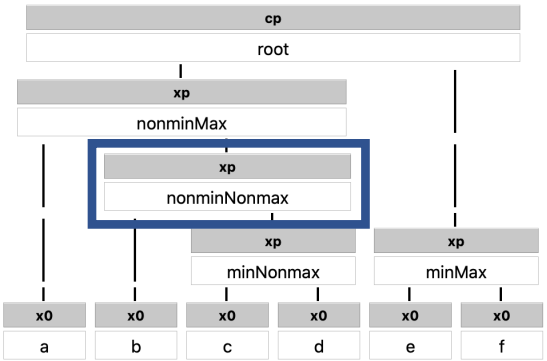

Figure 24. Syntactic input: [b c d] is [-Max, -Min]

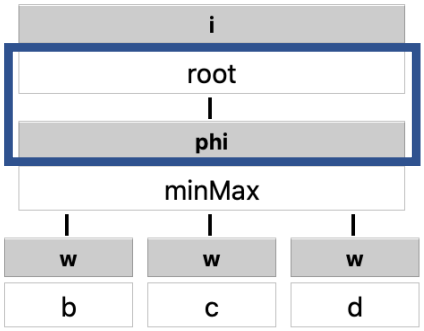

Figure 25. ptree19, (b c d) is [+Max, +Min]

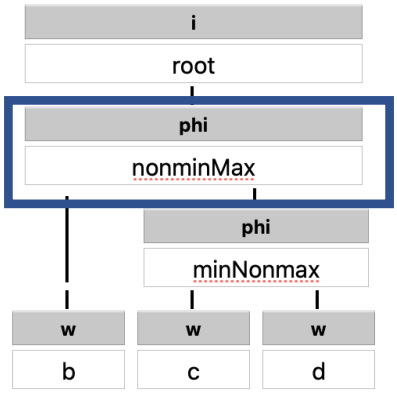

Figure 26. ptree20, (b c d) is [+Max, -Min]

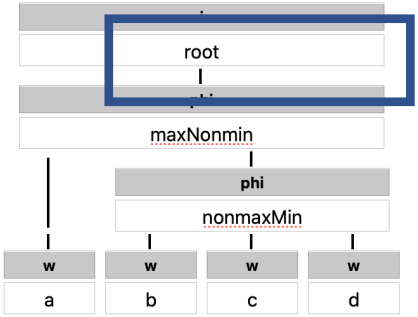

Figure 27. ptree21, (b c d) is [-Max, +Min]

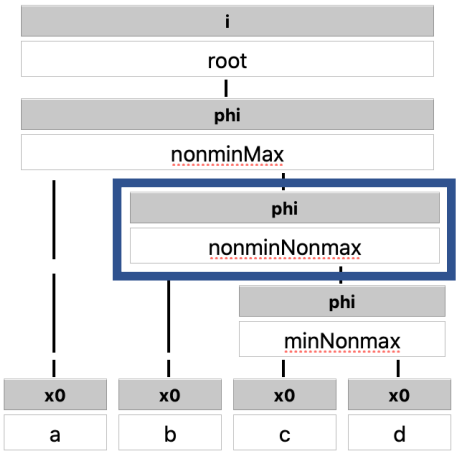

Figure 28. ptree22, (b c d) is [-Max, -Min]

Tableau 7. Expected violations for MatchSP(XP[-Max,-Min]), all possible prosody specifications

[a [b [c d]]] [e f] -MaxS -MinS	MXP -MaxS -MinS +MaxP	MXP -MaxS -MinS -MaxP	MXP -MaxS -MinS +MinP	MXP -MaxS -MinS +MaxP +MinP	MXP -MaxS -MinS -MaxP +MinP	MXP -MaxS -MinS -MinP	MXP -MaxS -MinS +MaxP -MinP	MXP -MaxS -MinS -MaxP -MinP
(a **(**b (c d)**)**) **-MaxP** **-MinP**	1	0	1	1	1	0	1	0
(a **(**b c d**)**) **-MaxP** **+MinP**	1	0	0	1	0	1	1	1
(b (c d)**)** **+MaxP** **-MinP**	0	1	1	1	1	0	0	1
(b c d**)** **+MaxP** **+MinP**	0	1	0	0	1	1	1	1
Mismatch	1	1	1	1	1	1	1	1

2.6 Syntax: -Max, Any Min

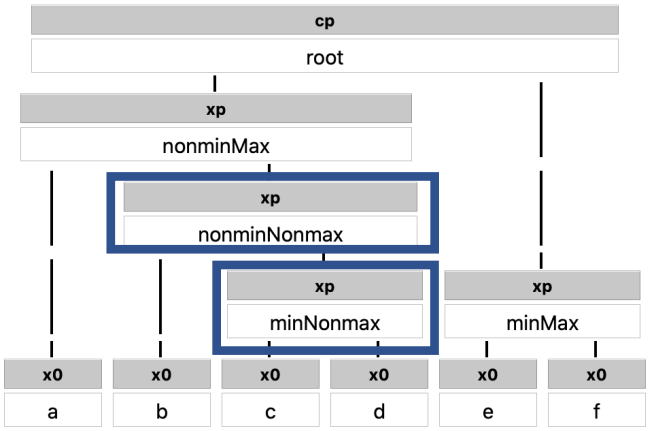

Figure 29. Syntactic input: [b c d], [c d] are [-Max]

Issue with nested XPs: The visibility setting [-Max, Any Min] allows for the possibility that the XPs targeted by Match are nested. When nested XPs must be matched to, e.g., $\varphi^{[+Max]}$, there will be at least one violation. For instance, it cannot be the case that both [b c d] and [c d] are mapped to a maximal φ.

Because of the nesting issue, I have focused on showing all 4 prosodic trees for the highest string targeted by MatchSP($XP^{[-Max]}$), which is [b c d]. *ptree19-22* are re-used, because those prosodic trees also targeted the XP [b c d]. However, both [b c d] and [c d] factor into the violation count.

Tableau 8. Expected violations for MatchSP($XP^{[-Max]}$), all possible prosody specifications

[a [b [c d]]] [e f] -MaxS	MXP -MaxS +MaxP	MXP -MaxS -MaxP	MXP -MaxS +MinP	MXP -MaxS +MaxP +MinP	MXP -MaxS -MaxP +MinP	MXP -MaxS -MinP	MXP -MaxS +MaxP -MinP	MXP -MaxS -MaxP -MinP
(a (b (c d))) **-MaxP** | **-MaxP** **-MinP** | **+MinP**	2	0	1	2	1	1	2	1
(a (b c d)) **-MaxP** | **N/A** **+MinP** | **N/A**	2	1	1	2	1	2	2	2
(b (c d)) **+MaxP** | **-MaxP** **-MinP** | **+MinP**	1	1	1	2	1	1	1	2
(b c d) **+MaxP** | **N/A** **+MinP** | **N/A**	1	2	1	1	2	2	2	2
Mismatch	2	2	2	2	2	2	2	2

2.7 Syntax: Any Max, +Min

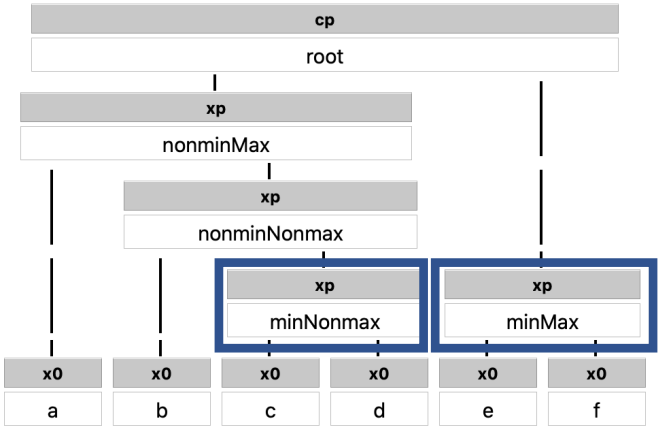

Figure 29. Syntactic input: [c d], [e f] are [+Min]

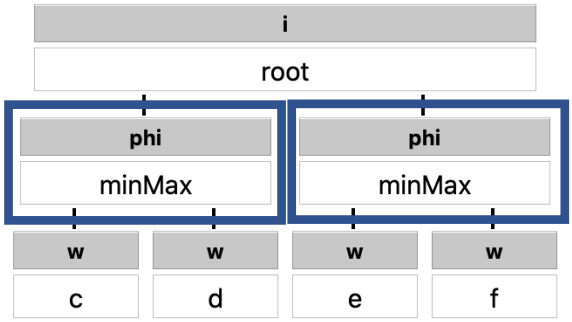

Figure 30. ptree23, (c d), (ef) are [+Max, +Min]

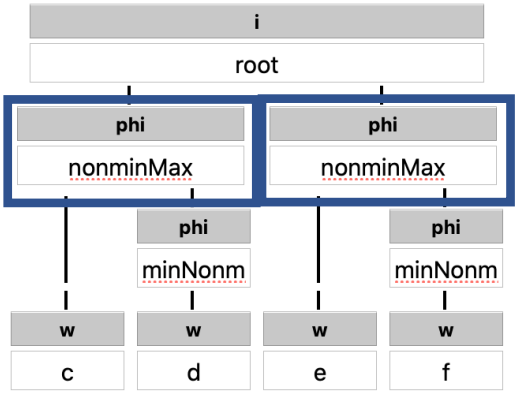

Figure 31. ptree24, (c d), (ef) are [+Max, -Min]

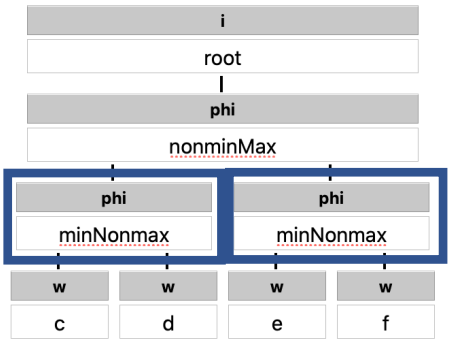

Figure 32. ptree25, (c d), (ef) are [-Max, +Min]

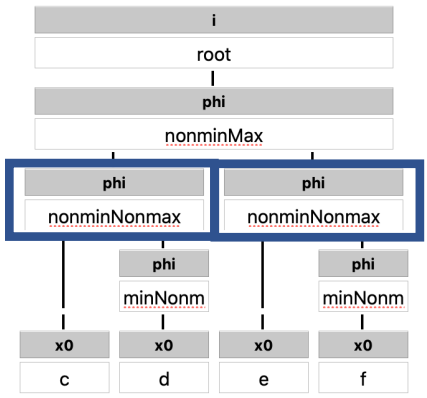

Figure 33. ptree26, (c d), (ef) are [-Max, -Min]

Tableau 9. Expected violations for MatchSP(XP[+Min]), all possible prosody specifications

[a [b [c d]]] [e f] +MinS	MXP +MinS +MaxP	MXP +MinS -MaxP	MXP +MinS +MinP	MXP +MinS +MaxP +MinP	MXP +MinS -MaxP +MinP	MXP +MinS -MinP	MXP +MinS +MaxP -MinP	MXP +MinS -MaxP -MinP
((c (d)) (e (f))) **-MaxP -MinP**	2	0	2	2	2	0	2	0
((c d) (e f)) **-MaxP +MinP**	2	0	0	2	0	2	2	2
(c (d)) (e (f)) **+MaxP -MinP**	0	2	2	2	2	0	0	2
(c d) (e f) **+MaxP +MinP**	0	2	0	0	2	2	2	2
Mismatch	2	2	2	2	2	2	2	2

2.8 Syntax: Any Max, - Min

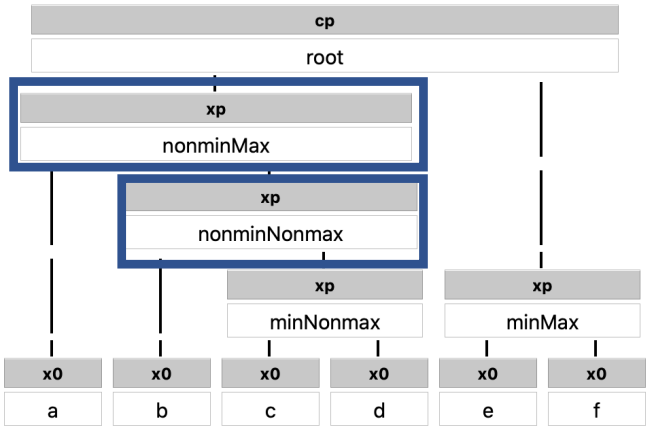

Figure 34. Syntactic input: [a b c d], [b c d] are [-Min]

Issue with nested XPs: The visibility setting [Any Max, -Min] allows for the possibility that the XPs targeted by Match are nested. When nested XPs must be matched to, e.g., φ[+Max], there will be at least one violation. For instance, it cannot be the case that both [a b c d] and [b c d] are mapped to a maximal φ.

Because of the nesting issue, I have focused on showing all 4 prosodic trees for the highest string targeted by MatchSP(XP[-Min]), which is [a b c d]. *ptree 7-10* are re-used, because those prosodic trees also targeted the XP [a b c d]. However, both [a b c d] and [b c d] factor into the violation count.

Tableau 10. Expected violations for MatchSP(XP$^{[-Min]}$), all possible prosody specifications

[a **[**b [c d]**]]** [e f] -MinS	MXP -MinS +MaxP	MXP -MinS -MaxP	MXP -MinS +MinP	MXP -MinS +MaxP +MinP	MXP -MinS -MaxP +MinP	MXP -MinS -MinP	MXP -MinS +MaxP -MinP	MXP -MinS -MaxP -MinP
((a (b c d)) e) **-MaxP** | **-MaxP** **-MinP** | **+MinP**	2	0	1	2	1	1	2	1
((a b c d) e) **-MaxP** | **N/A** **+MinP** | **N/A**	2	1	1	2	1	2	2	2
(a (b c d)) **+MaxP** | **-MaxP** **-MinP** | **+MinP**	1	1	1	2	1	1	1	2
(a b c d) **+MaxP** | **N/A** **+MinP** | **N/A**	1	2	1	1	2	2	2	2
Mismatch	2	2	2	2	2	2	2	2