Informative losers and ranking arguments

1. Finding valid ranking arguments: Comparative tableau format

(1) The **comparative tableau** format is a useful way to find cases of constraint conflict, and therefore to identify valid ranking arguments

- Consider again our tableau for /faslu/ [fas.lu] ‘his term’ in Cairene Arabic:

/faslu/ ‘his term’	NoOnsetCluster	NoDeletion	NoEpenthesis	NoCoda
→ a. fas.lu				*
b. fa.slu	* W			L
c. fa.lu		* W		L
d. fa.si.lu			* W	L

- We have added “winner preferring/loser preferring” notation (“W/L marks” for short) to this tableau, making it a **comparative tableau**

(2) To add W/L marks to a tableau, compare each loser in turn with the winner

(a) Take loser #1 (here, candidate (b), *[fa.slu]) and constraint #1 (here, NoOnsetCluster)

- Does this constraint prefer loser #1 over the winner? If so, put an **L mark** in the cell for loser #1 and constraint #1
- If this constraint prefers the winner instead, put a **W mark** in that cell
- If loser #1 and the winner are treated the same by this constraint, put no L or W

(b) Repeat with all other constraints and all other losing candidates

(c) Once you have finished, there should be at least one W mark in every row (*except the row for the winner,* where there are *no* L/W marks).

- When you have W and L marks in the same row, you have a case of **constraint conflict**, as needed in order to make a valid ranking argument
- If there is a loser with no W marks in its row, that loser is currently winning! You need to **add another constraint** that will prefer the winner over this loser

(3) Converting the information from L/W marks into a **constraint ranking**

(a) Any case where some constraint prefers the loser is crucial: if that constraint is ranked too high, then the loser that it prefers will (incorrectly) be chosen

(b) So, again looking at each loser in turn: ***every*** constraint with an L mark must be dominated by ***at least one*** constraint with a W mark (for the same loser)

(c) Keep track of the ranking relationships that you discover for each loser in turn, and in the end, combine them all into one consistent ranking

(4) Try it for yourself:

What constraint rankings are motivated by the Cairene Arabic tableau above?

(5) A more complicated example

- All lines in the tableau are dotted, because we haven't figured out the ranking yet

input	A	B	C	D	rankings?
→ winner					
loser #1		L	L	W	D » B ***and*** D » C
loser #2	L		W	W	C » A ***or*** D » A
loser #3	W		L		A » C

(a) We can simplify "{ C » A or D » A } and A » C" as "D » A and A » C"

(b) So, the rankings that are motivated here are: D » B, D » C, D » A, and A » C

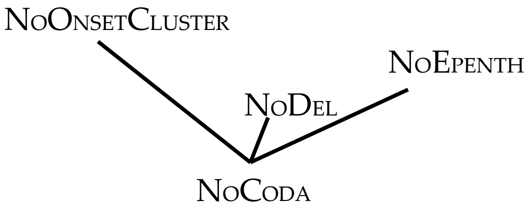
2. Hasse diagrams

(6) Once we are talking about more than two or three constraints, the clearest way to show their ranking relationships is with a Hasse diagram

(a) A Hasse diagram is a type of tree diagram

(b) If one constraint is drawn higher than another, and the two are connected with a vertical line, this represents a claim that the higher one dominates the lower one

(c) Applying this to Cairene Arabic:



- NoCoda is dominated by all three of the other constraints
- No ranking can be determined among the other three constraints

3. Informative losers

(7) If we are trying to propose an analysis (find a constraint ranking) for one particular language, what information do we start with? What else do we need to find?

(a) We know what the **output** is—this is the surface form observed in the language

(b) After doing phonological analysis as usual, we have a proposal for the **input** (UR)

(c) But in order to find evidence for constraint rankings, we need to add **losing candidates** to our tableau (we need L/W marks showing constraint conflict)

(d) Remember what we saw in the "money vs. love" example...

- We have to be careful to choose **informative losers**, which set up situations of **constraint conflict**, in order to argue for valid constraint rankings

(8) Look at the **winning candidate** to see what constraints it **violates**

- Now try some losers that satisfy these constraints (by violating other constraints)

(9) Other approaches: A loser *might* be informative if any of the following are true:

(a) The loser is known to be a winner in **other languages**

(b) The loser is the candidate that would be preferred by one of the constraints you are already discussing

(c) The loser is the **faithful** candidate, which doesn't change anything from the UR

(d) The loser otherwise illustrates a point you want to make about the language pattern or about the constraints you are discussing

(10) The importance of thinking about the **faithful** candidate

(a) If the faithful candidate is the *winner*, you can show that each markedness constraint that it violates is dominated by all relevant faithfulness constraints, leaving "no way out" from the markedness violation.

(b) If the faithful candidate is a *loser*, you know that the markedness constraint(s) that it violates is/are higher than at least one faithfulness constraint, because (at least) the lowest-ranked relevant faithfulness constraint is violated in the winner

(11) Watch out for these points in finding informative losers:

(a) A loser that differs greatly from the winner probably violates so *many* constraints that it becomes difficult to determine which of its violations are the relevant ones

- Therefore, a useful strategy is to look at losers that differ from the winner as **minimally** as possible—for example, one segment has been added or deleted; or one segment has been put in a different syllable position; or one segment has had a property changed; etc.

(b) An informative loser will always do **better** than the winner on at least one constraint—if not, it is **guaranteed to lose** under *all* constraint rankings

- Remember that we need a case of **constraint conflict** to prove a ranking; if all constraints favor the winner, there is no conflict