Computational Phonology - Class 4

Jeffrey Heinz (Instructor)
Jon Rawski (TA)

LSA Summer Institute
UC Davis
July 03, 2019

So far

1. We studied the successor and precedence model for words, both with and without phonological features.
2. We learned how to express functions $f : \Sigma^* \to \{\texttt{true}, \texttt{false}\}$ (constraints) in Monadic Second Order Logic with these models.
3. We learned about semirings and how to use them with MSO logic to express functions that maps strings to natural/real numbers.

Today

1. MSO Definable Transformations

Part I

MSO-Definable Transformations

Pedagogical strategy

Teach by example

1. Word-final obstruent devoicing (e.g. Russian).
2. [a]-Epenthesis to avoid word-final codas (e.g. Malagasy)
3. Total reduplicaton.

Functions

1. They have a *pre-image* (the set of structures to which the function applies; c.f. *domain*)
2. They have a *image* (the set of structures which the functions maps to; c.f. *co-domain*)
3. Determine which elements of the pre-image are mapped to which elements of the image.

MSO-definable transformations

Several items are needed.

1. Model signature for input structures.

(Courcelle 1994, Engelfriedt and Courcelle 2001, 2011)

MSO-definable transformations

Several items are needed.

1. Model signature for input structures.
2. Model signature for output structures.

(Courcelle 1994, Engelfriedt and Courcelle 2001, 2011)

MSO-definable transformations

Several items are needed.

1. Model signature for input structures.
2. Model signature for output structures.
3. One *domain* formula with no free variables.

(Courcelle 1994, Engelfriedt and Courcelle 2001, 2011)

MSO-definable transformations

Several items are needed.

1. Model signature for input structures.
2. Model signature for output structures.
3. One *domain* formula with no free variables.
4. A copyset C of numbers (like $\{1\}$ or $\{1, 2\}$).

(Courcelle 1994, Engelfriedt and Courcelle 2001, 2011)

MSO-definable transformations

Several items are needed.

1. Model signature for input structures.
2. Model signature for output structures.
3. One *domain* formula with no free variables.
4. A copyset C of numbers (like $\{1\}$ or $\{1, 2\}$).
5. $|C|$ *licensing* formulas with one free variable.

(Courcelle 1994, Engelfriedt and Courcelle 2001, 2011)

MSO-definable transformations

Several items are needed.

1. Model signature for input structures.
2. Model signature for output structures.
3. One *domain* formula with no free variables.
4. A copyset C of numbers (like $\{1\}$ or $\{1, 2\}$).
5. $|C|$ *licensing* formulas with one free variable.
6. $|C|^n$ formulas of n free variables for each relation of arity n in the signature of the output model.
    - So $|C|$ formulas with one free variable for each unary relation in the signature of the output model,
    - and $|C|^2$ formulas with two free variables for each binary relation in the signature of the output model, and so on.

(Courcelle 1994, Engelfriedt and Courcelle 2001, 2011)

MSO-definable transformations

Several items are needed.

1. Model signature for input structures.
2. Model signature for output structures.
3. One *domain* formula with no free variables.
4. A copyset C of numbers (like $\{1\}$ or $\{1, 2\}$).
5. $|C|$ *licensing* formulas with one free variable.
6. $|C|^n$ formulas of n free variables for each relation of arity n in the signature of the output model.
   - So $|C|$ formulas with one free variable for each unary relation in the signature of the output model,
   - and $|C|^2$ formulas with two free variables for each binary relation in the signature of the output model, and so on.

That's it!

(Courcelle 1994, Engelfriedt and Courcelle 2001, 2011)

Example: Word-final obstruent devoicing

$$
\begin{aligned}
\mathcal{M}_{input} &\stackrel{\text{def}}{=} ??\\
\mathcal{M}_{output} &\stackrel{\text{def}}{=} ??\\
\varphi_{domain} &\stackrel{\text{def}}{=} ??\\
C &\stackrel{\text{def}}{=} ??\\
\varphi_{licensing} &\stackrel{\text{def}}{=} ??
\end{aligned}
$$

And for each relation R in $\mathcal{M}_{output}$:
A formula of MSO($\mathcal{M}_{input}$) with as many free variables as the arity of R.

- So 1 free variable for unary relations,
- and 2 free variable for binary relations
- ...

Example: Word-final obstruent devoicing

$$\begin{aligned}
\mathcal{M}_{input} &\stackrel{\text{def}}{=} \mathcal{M}^{\triangleleft}_{features}\\
\mathcal{M}_{output} &\stackrel{\text{def}}{=} ??\\
\varphi_{domain} &\stackrel{\text{def}}{=} ??\\
C &\stackrel{\text{def}}{=} ??\\
\varphi_{licensing} &\stackrel{\text{def}}{=} ??
\end{aligned}$$

And for each relation R in $\mathcal{M}_{output}$:
A formula of MSO($\mathcal{M}_{input}$) with as many free variables as the arity of R.

- So 1 free variable for unary relations,
- and 2 free variable for binary relations
- …

Example: Word-final obstruent devoicing

$$
\begin{aligned}
\mathcal{M}_{input} &\stackrel{\text{def}}{=} \mathcal{M}^{\triangleleft}_{features} \\
\mathcal{M}_{output} &\stackrel{\text{def}}{=} \mathcal{M}^{\triangleleft}_{features} \\
\varphi_{domain} &\stackrel{\text{def}}{=} ?? \\
C &\stackrel{\text{def}}{=} ?? \\
\varphi_{licensing} &\stackrel{\text{def}}{=} ??
\end{aligned}
$$

And for each relation R in $\mathcal{M}_{output}$:
A formula of MSO($\mathcal{M}_{input}$) with as many free variables as the arity of R.

- So 1 free variable for unary relations,
- and 2 free variable for binary relations
- …

Example: Word-final obstruent devoicing

$$
\begin{aligned}
\mathcal{M}_{input} &\stackrel{\text{def}}{=} \mathcal{M}^{\triangleleft}_{features} \\
\mathcal{M}_{output} &\stackrel{\text{def}}{=} \mathcal{M}^{\triangleleft}_{features} \\
\varphi_{domain} &\stackrel{\text{def}}{=} \texttt{true} \\
C &\stackrel{\text{def}}{=} ?? \\
\varphi_{licensing} &\stackrel{\text{def}}{=} ??
\end{aligned}
$$

And for each relation R in $\mathcal{M}_{output}$:
A formula of MSO($\mathcal{M}_{input}$) with as many free variables as the arity of R.

- So 1 free variable for unary relations,
- and 2 free variable for binary relations
- …

Example: Word-final obstruent devoicing

$$
\begin{aligned}
\mathcal{M}_{input} &\overset{\text{def}}{=} \mathcal{M}^{\triangleleft}_{features} \\
\mathcal{M}_{output} &\overset{\text{def}}{=} \mathcal{M}^{\triangleleft}_{features} \\
\varphi_{domain} &\overset{\text{def}}{=} \texttt{true} \\
C &\overset{\text{def}}{=} \{1\} \\
\varphi_{licensing} &\overset{\text{def}}{=} ??
\end{aligned}
$$

And for each relation R in $\mathcal{M}_{output}$:
A formula of MSO($\mathcal{M}_{input}$) with as many free variables as the arity of R.

- So 1 free variable for unary relations,
- and 2 free variable for binary relations
- ...

Example: Word-final obstruent devoicing

$$
\begin{aligned}
\mathcal{M}_{input} &\stackrel{\text{def}}{=} \mathcal{M}^{\triangleleft}_{features} \\
\mathcal{M}_{output} &\stackrel{\text{def}}{=} \mathcal{M}^{\triangleleft}_{features} \\
\varphi_{domain} &\stackrel{\text{def}}{=} \texttt{true} \\
C &\stackrel{\text{def}}{=} \{1\} \\
\varphi_{licensing} &\stackrel{\text{def}}{=} \texttt{true}
\end{aligned}
$$

And for each relation R in $\mathcal{M}_{output}$:
A formula of MSO($\mathcal{M}_{input}$) with as many free variables as the arity of R.

- So 1 free variable for unary relations,
- and 2 free variable for binary relations
- …

Example: Word-final obstruent devoicing

Formulas for the relations in $\mathcal{M}_{output}$

- **Example formula with 1 free variable for the unary relation `vocalic`:**

$$\underbrace{\phi_{\texttt{vocalic}}(x)}_{\text{Does x have the feature \texttt{vocalic} in the output model?}} \overset{\text{def}}{=} \underbrace{\texttt{vocalic(x)}}_{\text{Evaluate with respect to the input model.}}$$

Example: Word-final obstruent devoicing

Formulas for the relations in $\mathcal{M}_{output}$

- **Example formula with 1 free variable for the unary relation `vocalic`:**

$$\underbrace{\phi_{\texttt{vocalic}}(x)}_{\text{Does x have the feature vocalic in the output model?}} \stackrel{\text{def}}{=} \underbrace{\texttt{vocalic(x)}}_{\text{Evaluate with respect to the input model.}}$$

- **Example formula with 2 free variables for the binary relation $\triangleleft$:**

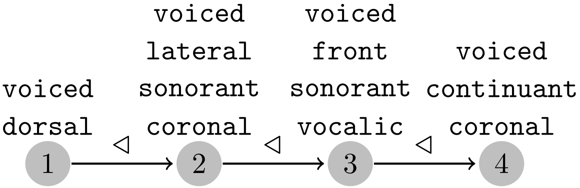
$$\underbrace{\phi_{\triangleleft}(x, y)}_{\text{Do x and y in the output model stand in the successor relation?}} \stackrel{\text{def}}{=} \underbrace{x \triangleleft y}_{\text{Evaluate with respect to the input model.}}$$

Example: Word-final obstruent devoicing

$$\varphi_{domain} \overset{\text{def}}{=} \texttt{true} \tag{1}$$

Consider model for the input *gliz*. So:

- $\mathcal{D} = \{1, 2, 3, 4\}$
- $\triangleleft = \{(1,2), (2,3), (3,4)\}$
- $\texttt{sonorant} = \{2, 3\}$
- $\texttt{voiced} = \{1, 2, 3, 4\}$
- $\ldots$



Example: Word-final obstruent devoicing

Building the domain of the output structure

$$C \stackrel{\text{def}}{=} \{1\} \tag{2}$$

$$\varphi_{license}(x) \stackrel{\text{def}}{=} \texttt{true} \tag{3}$$

1 2 3 4 (Copy set 1)

Example: Word-final obstruent devoicing

Adding the binary relation $\lhd$

$$\varphi_{\lhd}(x, y) \overset{\text{def}}{=} x \lhd y \tag{4}$$

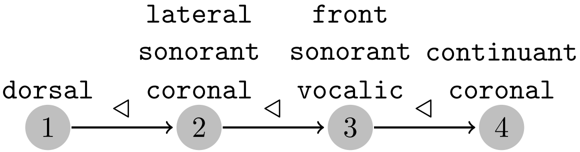
$$1 \xrightarrow{\lhd} 2 \xrightarrow{\lhd} 3 \xrightarrow{\lhd} 4$$

If $\varphi_{\lhd}(x, y)$ evaluates to true for the assignment $x \mapsto e_1, y \mapsto e_2$ then there (e_1, e_2) stands in the successor $(\lhd)$ relation in the output structure. The formula is evaluated w.r.t. the *input* structure.

Example: Word-final obstruent devoicing

Adding unary relations for all $\texttt{feature} \in \mathcal{F}$ (except $\texttt{voiced}$.)

$$\text{for all } \texttt{feature} \neq \texttt{voiced}: \quad \varphi_{\texttt{feature}}(x) \stackrel{\text{def}}{=} \texttt{feature}(x) \qquad (5)$$

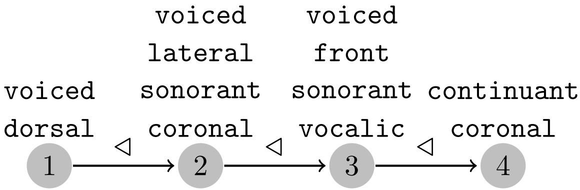


If $\varphi_{\texttt{feature}}(x)$ evaluates to true for the assignment $x \mapsto e$ then position e has the property $\texttt{feature}$ in the output structure. The formula is evaluated w.r.t. the *input* structure.

Example: Word-final obstruent devoicing

Adding the unary relation `voiced`.

$$\varphi_{\texttt{voiced}}(x) \overset{\text{def}}{=} \texttt{voiced}(x) \wedge \neg\big(\texttt{last}(x) \wedge \texttt{obstruent}(x)\big) \tag{6}$$



If $\varphi_{\texttt{voiced}}(x)$ evaluates to true for the assignment $x \mapsto e$ then position e has the property `feature` in the output structure. The formula is evaluated w.r.t. the *input* structure.

Example: Word-final obstruent devoicing

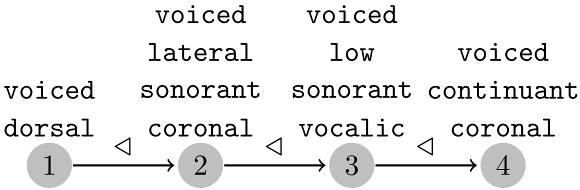
That's it!

$$\mathcal{M}_{input} = \mathcal{M}_{output} = \mathcal{M}^{\triangleleft}_{features}$$

$$\begin{aligned}
\varphi_{domain} &\stackrel{\text{def}}{=} \texttt{true} && (1)\\
C &\stackrel{\text{def}}{=} \{1\} && (2)\\
\varphi_{license}(x) &\stackrel{\text{def}}{=} \texttt{true} && (3)\\
\varphi_{\triangleleft}(x,y) &\stackrel{\text{def}}{=} x \triangleleft y && (4)\\
\varphi_{\texttt{feature}}(x) &\stackrel{\text{def}}{=} \texttt{feature}(x) && (5)\\
\varphi_{\texttt{voiced}}(x) &\stackrel{\text{def}}{=} \texttt{voiced}(x) \wedge \neg\big(\texttt{last}(x) \wedge \texttt{obstruent}(x)\big) && (6)
\end{aligned}$$

Example: [a]-Epenthesis to avoid word-final codas

glaz



Let this process apply to all structures.

$$\varphi_{domain} \stackrel{\text{def}}{=} \texttt{true} \tag{1}$$

Some notes

Formulae can be defined in any order, as long as they are all well-defined.

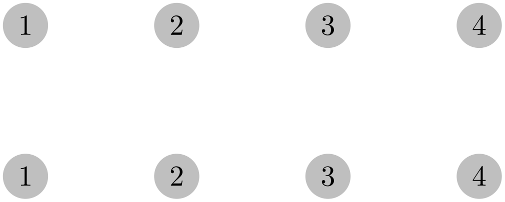
Together, the copy set C and the licensing formula determine the elements in the domain of the output structure.

- A copyset larger than size 1 is only needed if output structures can be larger than the input structures. If not, set $C = \{1\}$. (*regulates epenthesis, copying, 'growth'*)
- The licensing formulae are also only needed if the domain of the output structures are a different size from the domain of the input structures. (*regulates deletion*)

Example: [a]-Epenthesis to avoid word-final codas

Building the domain of the output structure

$$C \stackrel{\text{def}}{=} \{1, 2\} \tag{2}$$



(Copy set 1)

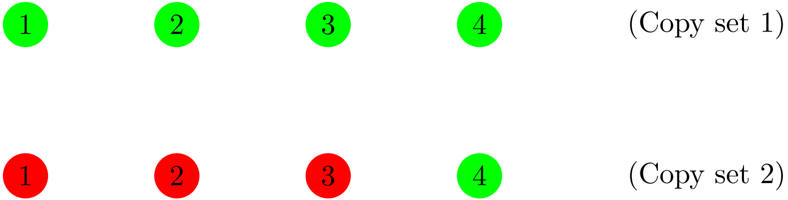
(Copy set 2)

Example: [a]-Epenthesis to avoid word-final codas

Licensing the elements we wish to keep.

$$\varphi^1_{license}(x) \stackrel{\text{def}}{=} \texttt{true} \tag{3}$$

$$\varphi^2_{license}(x) \stackrel{\text{def}}{=} \texttt{last}(x) \wedge \texttt{cons}(x) \tag{4}$$



Example: [a]-Epenthesis to avoid word-final codas

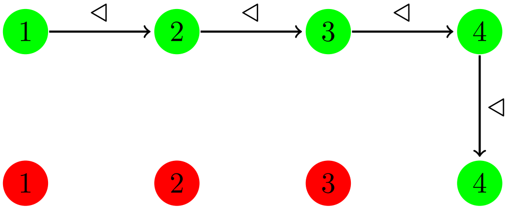
Adding the binary relation $\triangleleft$.

$$\varphi_{\triangleleft}^{1,1}(x,y) \stackrel{\text{def}}{=} x \triangleleft y \tag{5}$$

$$\varphi_{\triangleleft}^{1,2}(x,y) \stackrel{\text{def}}{=} \texttt{last}(x) \wedge \texttt{last}(y) \tag{6}$$

$$\varphi_{\triangleleft}^{2,1}(x,y) \stackrel{\text{def}}{=} \texttt{false} \tag{7}$$

$$\varphi_{\triangleleft}^{2,2}(x,y) \stackrel{\text{def}}{=} \texttt{false} \tag{8}$$



(Copy set 1)

(Copy set 2)

Example: [a]-Epenthesis to avoid word-final codas

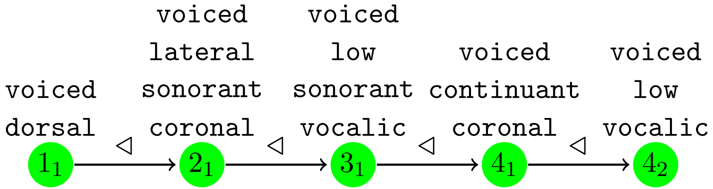
Adding unary relations.

$$\varphi^1_{\texttt{feature}}(x) \stackrel{\text{def}}{=} \texttt{feature}(x) \quad \text{(for all features)} \tag{9}$$

$$\varphi^2_{\texttt{vocalic}}(x) \stackrel{\text{def}}{=} \texttt{last}(x) \tag{10}$$

$$\varphi^2_{\texttt{low}}(x) \stackrel{\text{def}}{=} \texttt{last}(x) \tag{11}$$

$$\ldots \tag{12}$$

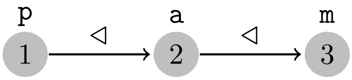


Example: [a]-Epenthesis to avoid word-final codas

That's it!

Example: Total Reduplication

pam



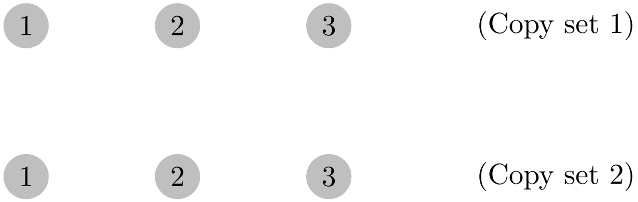
Let the transformation apply to all structures.

$$\varphi_{domain} \overset{\text{def}}{=} \texttt{true} \tag{1}$$

Example: Total Reduplication

Building the domain of the output structure

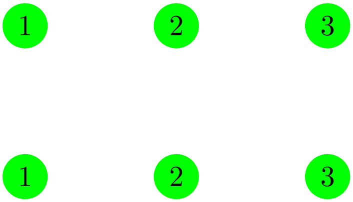
$$C \stackrel{\text{def}}{=} \{1, 2\} \tag{2}$$



Example: Total Reduplication

Licensing the elements we wish to keep.

$$\varphi^1_{license}(x) \overset{\text{def}}{=} \varphi^2_{license}(x) \overset{\text{def}}{=} \texttt{true} \tag{3}$$



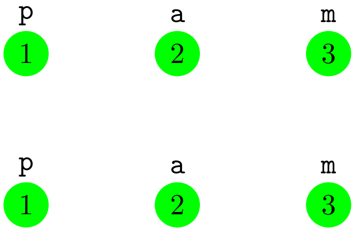
(Copy set 1)

(Copy set 2)

Example: Total Reduplication

Adding unary relations.

$$\varphi^1_{\texttt{feature}}(x) \overset{\text{def}}{=} \varphi^2_{\texttt{feature}}(x) \overset{\text{def}}{=} \texttt{feature}(x) \quad \text{(for all features)} \tag{4}$$



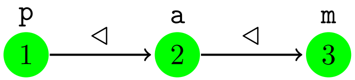
(Copy set 1)

(Copy set 2)

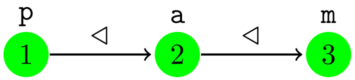
Example: Total Reduplication

Adding the binary relation $\lhd$ (part 1)

$$\varphi_{\lhd}^{1,1}(x,y) \overset{\text{def}}{=} \varphi_{\lhd}^{2,2}(x,y) \overset{\text{def}}{=} x \lhd y \tag{5}$$



(Copy set 1)



(Copy set 2)

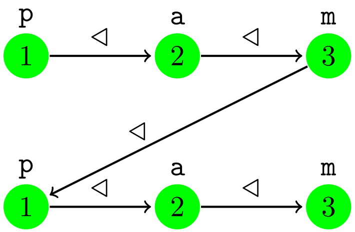
Example: Total Reduplication

Adding the binary relation $\triangleleft$ (part 2)

$$\varphi_{\triangleleft}^{1,1}(x,y) \stackrel{\text{def}}{=} \varphi_{\triangleleft}^{2,2}(x,y) \stackrel{\text{def}}{=} x \triangleleft y \tag{6}$$

$$\varphi_{\triangleleft}^{1,2}(x,y) \stackrel{\text{def}}{=} \texttt{last}(x) \wedge \texttt{first}(y) \tag{7}$$

$$\varphi_{\triangleleft}^{2,1}(x,y) \stackrel{\text{def}}{=} \texttt{false} \tag{8}$$



(Copy set 1)

(Copy set 2)

Example: Total Reduplication

That's it!

Summary

1. MSO-definable transductions are specified with a copyset and several formula.
    1. The domain formula determines the pre-image (the structures the transformation applies to).
    2. The copyset and licensing formula determined the elements of the output structure.
    3. The other formula specify the relations among elements in the output structure.
2. The input models and the output models can have different signatures!
3. This is NOT a theory of phonology; but a precise description language for transformations. (Though theories CAN be stated within this framework.)

Homework 3

1. Write a MSO-definable tranduction for the phonological *process* of your choice.
   - Example: Write a MSO-definable transduction for intervocalic voicing. More specically, all consonants are voiced intervocalically.
   - Example: Write a MSO-definable transduction for [i]-prothesis before a word-initial consonant cluster. (For example, many Nepalese pronounce English [skul] "school" as [iskul].)
2. (optional/bonus) Let $\Sigma = \{a, b, c\}$. Write a MSO-definable transduction which *sorts* the letters in a string in alphabetic order. Examples:

$$\begin{array}{rcl} \text{bac} & \mapsto & \text{abc} \\ \text{cba} & \mapsto & \text{abc} \\ \text{bbabca} & \mapsto & \text{aabbbc} \\ & \dots & \end{array}$$

(Hint: Use a copy set equal to the size of Σ.)