Functional Arabic Morphology

Principles of Design

Otakar Smrž

Institute of Formal and Applied Linguistics
Faculty of Mathematics and Physics
Charles University in Prague

Prague, November 6, 2006

He will notify them about that through SMS messages, the Internet, and other means. سيخبرهم بذلك عن طريق الرسائل القصيرة والإنترنت وغيرها.

He will notify them about that through SMS messages, the Internet, and other means. سَيُخبِرُهُم بِذٰلِكَ عَن طَرِيقِ الرَّسَائِلِ القَصِيرَةِ وَالإِنتِرنِتِ وَغَيرِهَا.

String	Token	Token Tag	Buckwalter's M-Tags	Token Form	Token Gloss
سيخبرهم		F---------	FUT	*sa-*	will
		VIIA-3MS--	IV3MS+IV+IVSUFF_MOOD:I	*yu-ḫbir-u*	he-notify
		S----3MP4-	IVSUFF_DO:3MP	*-hum*	them
بذلك		P---------	PREP	*bi-*	about/by
		SD----MS--	DEM_PRON_MS	*ḏālika*	that
عن		P---------	PREP	*ʿan*	by/about
طريق		N-------2R	NOUN+CASE_DEF_GEN	*ṭarīq-i*	way-of
الرسائل		N-------2D	DET+NOUN+CASE_DEF_GEN	*ar-rasāʾil-i*	the-messages
القصيرة		A-----FS2D	DET+ADJ+NSUFF_FEM_SG+ +CASE_DEF_GEN	*al-qaṣīr-at-i*	the-short
والإنترنت		C---------	CONJ	*wa-*	and
		Z-------2D	DET+NOUN_PROP+ +CASE_DEF_GEN	*al-ʾinternet-i*	the-internet
وغيرها		C---------	CONJ	*wa-*	and
		FN------2R	NEG_PART+CASE_DEF_GEN	*ġayr-i*	other/not-of
		S----3FS2-	POSS_PRON_3FS	*-hā*	them

He will notify them about that through SMS messages, the Internet, and other means. سَيُخبِرُهُم بِذٰلِكَ عَن طَرِيقِ الرَّسَائِلِ القَصِيرَةِ وَالإِنتِرنِتِ وَغَيرِهَا.

String	Token	Token Tag	Buckwalter's M-Tags	Token Form	Token Gloss
سيخبرهم		F---------	FUT	*sa-*	will
		VIIA-3MS--	IV3MS+IV+IVSUFF_MOOD:I	*yu-ḫbir-u*	he-notify
		S----3MP4-	IVSUFF_DO:3MP	*-hum*	them
بذلك		P---------	PREP	*bi-*	about/by
		SD----MS--	DEM_PRON_MS	*ḏālika*	that
عن		P---------	PREP	*ʿan*	by/about
طريق		N-------2R	NOUN+CASE_DEF_GEN	*ṭarīq-i*	way-of
الرسائل		N-------2D	DET+NOUN+CASE_DEF_GEN	*ar-rasāʾil-i*	the-messages
القصيرة		A-----FS2D	DET+ADJ+NSUFF_FEM_SG+ +CASE_DEF_GEN	*al-qaṣīr-at-i*	the-short
والإنترنت		C---------	CONJ	*wa-*	and
		Z-------2D	DET+NOUN_PROP+ +CASE_DEF_GEN	*al-ʾinternet-i*	the-internet
وغيرها		C---------	CONJ	*wa-*	and
		FN------2R	NEG_PART+CASE_DEF_GEN	*ġayr-i*	other/not-of
		S----3FS2-	POSS_PRON_3FS	*-hā*	them

He will notify them about that through SMS messages, the Internet, and other means. سَيُخبِرُهُم بِذٰلِكَ عَن طَرِيقِ الرَّسَائِلِ القَصِيرَةِ وَالإِنتِرنِتِ وَغَيرِهَا.

String	Token	Token Tag	Buckwalter's M-Tags	Token Form	Token Gloss
سيخبرهم		F---------	FUT	*sa-*	will
		VIIA-3MS--	IV3MS+IV+IVSUFF_MOOD:I	*yu-ḫbir-u*	he-notify
		S----3MP4-	IVSUFF_DO:3MP	*-hum*	them
بذلك		P---------	PREP	*bi-*	about/by
		SD----MS--	DEM_PRON_MS	*ḏālika*	that
عن		P---------	PREP	*ʿan*	by/about
طريق		N-------2R	NOUN+CASE_DEF_GEN	*ṭarīq-i*	way-of
الرسائل		N-------2D	DET+NOUN+CASE_DEF_GEN	*ar-rasāʾil-i*	the-messages
القصيرة		A-----FS2D	DET+ADJ+NSUFF_FEM_SG+ +CASE_DEF_GEN	*al-qaṣīr-at-i*	the-short
والإنترنت		C---------	CONJ	*wa-*	and
		Z-------2D	DET+NOUN_PROP+ +CASE_DEF_GEN	*al-ʾinternet-i*	the-internet
وغيرها		C---------	CONJ	*wa-*	and
		FN------2R	NEG_PART+CASE_DEF_GEN	*ġayr-i*	other/not-of
		S----3FS2-	POSS_PRON_3FS	*-hā*	them

Outline

1 Introduction

Outline

1. Introduction
2. Morphological Theory
   - Incremental vs. Realizational
   - Lexical vs. Inferential
   - Functional Arabic Morphology

Outline

1. Introduction
2. Morphological Theory
   - Incremental vs. Realizational
   - Lexical vs. Inferential
   - Functional Arabic Morphology
3. Implementation Design
   - ElixirFM
   - Paradigms, parameters, ...
   - Elixir Lexicon
   - FM Generic

Outline

1. Introduction
2. Morphological Theory
   - Incremental vs. Realizational
   - Lexical vs. Inferential
   - Functional Arabic Morphology
3. Implementation Design
   - ElixirFM
   - Paradigms, parameters, ...
   - Elixir Lexicon
   - FM Generic
4. Extensions
   - Encode Arabic
   - MorphoTrees

Outline

1. Introduction
2. Morphological Theory
   - Incremental vs. Realizational
   - Lexical vs. Inferential
   - Functional Arabic Morphology
3. Implementation Design
   - ElixirFM
   - Paradigms, parameters, ...
   - Elixir Lexicon
   - FM Generic
4. Extensions
   - Encode Arabic
   - MorphoTrees
5. References

Outline

1. Introduction
2. Morphological Theory
   - Incremental vs. Realizational
   - Lexical vs. Inferential
   - Functional Arabic Morphology
3. Implementation Design
   - ElixirFM
   - Paradigms, parameters, ...
   - Elixir Lexicon
   - FM Generic
4. Extensions
   - Encode Arabic
   - MorphoTrees
5. References

Inflectional Morphology

Morphological theories can be classified along two dimensions (Stump 2001).

lexical association of word's morphosyntactic properties with affixes

Inflectional Morphology

Morphological theories can be classified along two dimensions (Stump 2001).

lexical	association of word's morphosyntactic properties with affixes
inferential	inflection is a result of operations on lexemes; morphosyntactic properties are expressed by the rules that relate the form in a given paradigm to the lexeme

Inflectional Morphology

Morphological theories can be classified along two dimensions (Stump 2001).

lexical association of word's morphosyntactic properties with affixes

inferential inflection is a result of operations on lexemes; morphosyntactic properties are expressed by the rules that relate the form in a given paradigm to the lexeme

incremental words acquire morphosyntactic properties only in connection with acquiring the inflectional exponents of those properties

Inflectional Morphology

Morphological theories can be classified along two dimensions (Stump 2001).

lexical	association of word's morphosyntactic properties with affixes
inferential	inflection is a result of operations on lexemes; morphosyntactic properties are expressed by the rules that relate the form in a given paradigm to the lexeme
incremental	words acquire morphosyntactic properties only in connection with acquiring the inflectional exponents of those properties
realizational	association of a set of properties with a word licenses the introduction of the exponents into the word's morphology

Inflectional Morphology

Morphological theories can be classified along two dimensions (Stump 2001).

lexical	association of word's morphosyntactic properties with affixes
inferential	inflection is a result of operations on lexemes; morphosyntactic properties are expressed by the rules that relate the form in a given paradigm to the lexeme

incremental	words acquire morphosyntactic properties only in connection with acquiring the inflectional exponents of those properties
realizational	association of a set of properties with a word licenses the introduction of the exponents into the word's morphology

Evidence favoring inferential–realizational theories over the others is given.

Extended Exponence

The morphosyntactic properties associated with an inflected word may exhibit extended exponence in that word's morphology. (Stump 2001:4)

Extended Exponence

The morphosyntactic properties associated with an inflected word may exhibit extended exponence in that word's morphology. (Stump 2001:4)

F---------	FUT	*sa-*	will
VIIA-3MS--	IV3MS+IV+IVSUFF_MOOD:I	*yu-ḫbir-u*	he-notify
S----3MP4-	IVSUFF_DO:3MP	*-hum*	them

Extended Exponence

The morphosyntactic properties associated with an inflected word may exhibit extended exponence in that word's morphology. (Stump 2001:4)

F---------	FUT	*sa-*	will
VIIA-3MS--	IV3MS+IV+IVSUFF_MOOD:I	*yu-ḫbir-u*	he-notify
S----3MP4-	IVSUFF_DO:3MP	*-hum*	them

Underdetermination

The morphosyntactic properties associated with an inflected word's individual inflectional markings may underdetermine the properties associated with the word as a whole. (Stump 2001:7)

Underdetermination

The morphosyntactic properties associated with an inflected word's individual inflectional markings may underdetermine the properties associated with the word as a whole. (Stump 2001:7)

طريق	N-------2R	NOUN+CASE_DEF_GEN	*ṭarīq-i*	way-of
الرسائل	N-------2D	DET+NOUN+CASE_DEF_GEN	*ar-rasāʾil-i*	the-messages
القصيرة	A-----FS2D	DET+ADJ+NSUFF_FEM_SG+ +CASE_DEF_GEN	*al-qaṣīr-at-i*	the-short

Underdetermination

The morphosyntactic properties associated with an inflected word's individual inflectional markings may underdetermine the properties associated with the word as a whole. (Stump 2001:7)

طريق	N-----FS2R	NOUN+CASE_DEF_GEN	*ṭarīq-i*	way-of
الرسائل	N-----FS2D	DET+NOUN+CASE_DEF_GEN	*ar-rasāʾil-i*	the-messages
القصيرة	A-----FS2D	DET+ADJ+NSUFF_FEM_SG+ +CASE_DEF_GEN	*al-qaṣīr-at-i*	the-short

Underdetermination

The morphosyntactic properties associated with an inflected word's individual inflectional markings may underdetermine the properties associated with the word as a whole. (Stump 2001:7)

طريق	N-------2R	NOUN+CASE_DEF_GEN	*ṭarīq-i*	way-of
الرسائل	N-----FP2D	DET+NOUN+CASE_DEF_GEN	*ar-rasāʾil-i*	the-messages
القصيرة	A-----FS2D	DET+ADJ+NSUFF_FEM_SG+ +CASE_DEF_GEN	*al-qaṣīr-at-i*	the-short

Nonconcatenative Inflection

There is no theoretically significant difference between concatenative and nonconcatenative inflection. (Stump 2001:9)

Nonconcatenative Inflection

There is no theoretically significant difference between concatenative and nonconcatenative inflection. *(Stump 2001:9)*

		أخبر	*ʾaḫbar-a*	to notify
سيخبرهم	F---------	FUT	*sa-*	will
	VIIA-3MS--	IV3MS+IV+IVSUFF_MOOD:I	*yu-ḫbir-u*	he-notify
	S----3MP4-	IVSUFF_DO:3MS	*-hum*	them
		رسالة	*risāl-at-un*	a message
الرسائل	N-------2D	DET+NOUN+CASE_DEF_GEN	*ar-rasāʾil-i*	the-messages

Nonconcatenative Inflection

There is no theoretically significant difference between concatenative and nonconcatenative inflection. *(Stump 2001:9)*

		أخبر *ʾaḫbar-a*		to notify
سيخبرهم	F---------	FUT	*sa-*	will
	VIIA-3MS--	IV3MS+IV+IVSUFF_MOOD:I	*yu-ḫbir-u*	he-notify
	S----3MP4-	IVSUFF_DO:3MS	*-hum*	them
		رسالة *risāl-at-un*		a message
الرسائل	N-------2D	DET+NOUN+CASE_DEF_GEN	*ar-rasāʾil-i*	the-messages

Unmotivated Choice

Exponence is the only association between inflectional markings and morphosyntactic properties. *(Stump 2001:11)*

`IV3MS+IV+IVSUFF_MOOD:I` ?? `IV3MS+IV+IVSUFF_MOOD:I` *yu-ḫbir-u*

Unmotivated Choice

Exponence is the only association between inflectional markings and morphosyntactic properties. *(Stump 2001:11)*

`IV3MS+IV+IVSUFF_MOOD:I` ?? `IV3MS+IV+IVSUFF_MOOD:I` *yu-ḫbir-u*

An uncompounded word's morphological form is not distinct from its phonological form. *(Stump 2001:12)*

`DET+ADJ+NSUFF_FEM_SG+CASE_DEF_GEN` *(al-(qaṣīr-at))-i* ?? *((al-qaṣīr)-at)-i*

Functional Arabic Morphology

Most computational models of Arabic morphology are lexical in nature. As they are not designed in connection with any syntax–morphology interface, their interpretation is destined to be incremental.

Functional Arabic Morphology

Most computational models of Arabic morphology are lexical in nature. As they are not designed in connection with any syntax–morphology interface, their interpretation is destined to be incremental.

Functional Arabic Morphology endorses the inferential–realizational views.

Functional Arabic Morphology

Most computational models of Arabic morphology are lexical in nature. As they are not designed in connection with any syntax–morphology interface, their interpretation is destined to be incremental.

Functional Arabic Morphology endorses the inferential–realizational views.

It re-establishes the system of inflectional and inherent morphosyntactic properties and distinguishes precisely the senses of their use in the grammar.

Functional Arabic Morphology

Most computational models of Arabic morphology are lexical in nature. As they are not designed in connection with any syntax–morphology interface, their interpretation is destined to be incremental.

Functional Arabic Morphology endorses the inferential–realizational views.

It re-establishes the system of inflectional and inherent morphosyntactic properties and distinguishes precisely the senses of their use in the grammar.

Definition of lexemes can include the derivational root and pattern information if appropriate. Modeling of the written language as well as spoken dialects is expected to be methodologically identical.

Outline

1. Introduction
2. Morphological Theory
   - Incremental vs. Realizational
   - Lexical vs. Inferential
   - Functional Arabic Morphology
3. Implementation Design
   - ElixirFM
   - Paradigms, parameters, ...
   - Elixir Lexicon
   - FM Generic
4. Extensions
   - Encode Arabic
   - MorphoTrees
5. References

ElixirFM

ElixirFM is a high-level implementation of Functional Arabic Morphology.

ElixirFM

ElixirFM is a high-level implementation of Functional Arabic Morphology.

ElixirFM uses the Functional Morphology library for Haskell and extends it.

ElixirFM

ElixirFM is a high-level implementation of Functional Arabic Morphology.

ElixirFM uses the Functional Morphology library for Haskell and extends it.

Morphology is modeled in terms of paradigms, grammatical categories, lexemes and word classes. The computation of analysis or generation is conceptually distinguished from the general-purpose linguistic model.

ElixirFM

ElixirFM is a high-level implementation of Functional Arabic Morphology.

ElixirFM uses the Functional Morphology library for Haskell and extends it.

Morphology is modeled in terms of paradigms, grammatical categories, lexemes and word classes. The computation of analysis or generation is conceptually distinguished from the general-purpose linguistic model.

The lexicon of ElixirFM is derived from the open-source Buckwalter lexicon and from the PADT annotations. It is redesigned in important respects.

Paradigms

A paradigm function is a function which, when applied to the root of a lexeme L paired with a set of morphosyntactic properties appropriate to L, determines the word form occupying the corresponding cell in L's paradigm. *(Stump 2001:32)*

Paradigms

A paradigm function is a function which, when applied to the root of a lexeme L paired with a set of morphosyntactic properties appropriate to L, determines the word form occupying the corresponding cell in L's paradigm. *(Stump 2001:32)*

```
paradigm   :: (Lexeme, Properties) -> WordForm
paradigm (l, ps) = ...
```

Paradigms

A paradigm function is a function which, when applied to the root of a lexeme L paired with a set of morphosyntactic properties appropriate to L, determines the word form occupying the corresponding cell in L's paradigm. *(Stump 2001:32)*

```
paradigm  :: (Lexeme, Properties) -> WordForm
paradigm (l, ps) = ...

paradigm'  :: Lexeme -> Properties -> WordForm
paradigm' l ps = paradigm (l, ps)
paradigm' l ps = (curry paradigm) l ps
paradigm' = curry paradigm

curry :: ((a, b) -> c) -> a -> b -> c
curry f x y = f (x, y)
```

Paradigms

A paradigm function is a function which, when applied to the root of a lexeme L paired with a set of morphosyntactic properties appropriate to L, determines the word form occupying the corresponding cell in L's paradigm. *(Stump 2001:32)*

```
paradigm  :: (Lexeme, Properties) -> WordForm
paradigm (l, ps) = ...

paradigm'  :: Lexeme -> Properties -> WordForm
paradigm' l ps = paradigm (l, ps)
paradigm' l ps = (curry paradigm) l ps
paradigm' = curry paradigm

curry :: ((a, b) -> c) -> a -> b -> c
curry f x y = f (x, y)
```

Paradigms

A paradigm function is a function which, when applied to the root of a lexeme L paired with a set of morphosyntactic properties appropriate to L, determines the word form occupying the corresponding cell in L's paradigm. *(Stump 2001:32)*

```
paradigm  :: (Lexeme, Properties) -> WordForm
paradigm (l, ps) = ...

paradigm'  :: Lexeme -> Properties -> WordForm
paradigm' l ps = paradigm (l, ps)
paradigm' l ps = (curry paradigm) l ps
paradigm' = curry paradigm

curry :: ((a, b) -> c) -> a -> b -> c
curry f x y = f (x, y)
```

Parameters

Instead of feature–value pairs for encoding the morphosyntactic properties (Stump 2001), we use enumerated values of distinct types. The use of data types is essential in the system (Forsberg and Ranta 2004).

Parameters

Instead of feature–value pairs for encoding the morphosyntactic properties (Stump 2001), we use enumerated values of distinct types. The use of data types is essential in the system (Forsberg and Ranta 2004).

```
data Person = First | Second | Third
    deriving (Eq, Enum)
```

Parameters

Instead of feature–value pairs for encoding the morphosyntactic properties (Stump 2001), we use enumerated values of distinct types. The use of data types is essential in the system (Forsberg and Ranta 2004).

```
data  Person  = First | Second | Third
    deriving (Eq, Enum)

data  Mood  = Indicative | Subjunctive
            | Jussive | Energetic
    deriving (Eq, Show, Enum)
```

Parameters

Instead of feature–value pairs for encoding the morphosyntactic properties (Stump 2001), we use enumerated values of distinct types. The use of data types is essential in the system (Forsberg and Ranta 2004).

```
data  Person  = First | Second | Third
    deriving (Eq, Enum)

data  Mood  = Indicative | Subjunctive
            | Jussive | Energetic
    deriving (Eq, Show, Enum)

data  ParaVerb  = VerbP Voice Person Gender Number
            | VerbI  Mood Voice Person Gender Number
            | VerbC                    Gender Number
    deriving Eq
```

Elixir Lexicon

(a) representation of the linguistic data in an abstract and extensible notation that encodes both orthography and phonology, and whose interpretation is customizable

Elixir Lexicon

(a) representation of the linguistic data in an abstract and extensible notation that encodes both orthography and phonology, and whose interpretation is customizable

(b) organization of the lexicon so that there is preferably no duplication of information and so that the lexicon can possibly be divided into separate units, as well as be interlinked with external modules

Elixir Lexicon

(a) representation of the linguistic data in an abstract and extensible notation that encodes both orthography and phonology, and whose interpretation is customizable

(b) organization of the lexicon so that there is preferably no duplication of information and so that the lexicon can possibly be divided into separate units, as well as be interlinked with external modules

(c) definition of such format of the lexicon so that editing and understanding the data is not inappropriately difficult, and using such data markup whose syntax is either lightweight, or can be edited/verified with some automatic tools, or both

FM Generic

The linguistic model and the data of the lexicon can be compiled into run-time applications or used as standalone libraries and resources.

FM Generic

The linguistic model and the data of the lexicon can be compiled into run-time applications or used as standalone libraries and resources.

FM Generic implements the compilation of morphological analyzers and generators (Forsberg and Ranta 2004). The method used for analysis is deterministic parsing with tries (Ljunglöf 2002).

FM Generic

The linguistic model and the data of the lexicon can be compiled into run-time applications or used as standalone libraries and resources.

FM Generic implements the compilation of morphological analyzers and generators (Forsberg and Ranta 2004). The method used for analysis is deterministic parsing with tries (Ljunglöf 2002).

FM Generic also provides functions for exporting and pretty-printing the linguistic model into XFST, Lexc, SQL, XML, LaTeX, ...

Outline

1. Introduction
2. Morphological Theory
   - Incremental vs. Realizational
   - Lexical vs. Inferential
   - Functional Arabic Morphology
3. Implementation Design
   - ElixirFM
   - Paradigms, parameters, ...
   - Elixir Lexicon
   - FM Generic
4. Extensions
   - Encode Arabic
   - MorphoTrees
5. References

Buckwalter Transliteration

يُولَدُ جَمِيعُ ٱلنَّاسِ أَحْرَارًا مُتَسَاوِينَ فِي ٱلْكَرَامَةِ وَٱلْحُقُوقِ. وَقَدْ وُهِبُوا عَقْلًا وَضَمِيرًا وَعَلَيْهِمْ أَنْ يُعَامِلَ بَعْضُهُمْ بَعْضًا بِرُوحِ ٱلْإِخَاءِ.

```
yuwladu jamiyEu  {ln~aAsi OaHoraArFA mutasaAwiyna fiy
{lokaraAmapi wa {loHuquwqi.  waqado wuhibuwA EaqolAF
waDamiyrFA waEalayohimo Oano yuEaAmila baEoDuhumo baEoDFA
biruwHi  {loIixaA'i.
```

يولد جميع الناس أحرارا متساوين في الكرامة والحقوق. وقد وهبوا عقلا وضميرا وعليهم أن يعامل بعضهم بعضا بروح الإخاء.

```
ywld jmyE AlnAs OHrArA mtsAwyn fy AlkrAmp wAlHqwq.  wqd
whbwA EqlA wDmyrA wElyhm On yEAml bEDhm bEDA brwH AlIxA'.
```

Notation of ArabTEX

يُولَدُ جَمِيعُ ٱلنَّاسِ أَحْرَارًا مُتَسَاوِينَ فِي ٱلْكَرَامَةِ وَٱلْحُقُوقِ. وَقَدْ وُهِبُوا عَقْلًا وَضَمِيرًا وَعَلَيْهِمْ أَنْ يُعَامِلَ بَعْضُهُمْ بَعْضًا بِرُوحِ ٱلْإِخَاءِ.

يولد جميع النّاس أحرارا متساوين في الكرامة والحقوق. وقد وهبوا عقلا وضميرا وعليهم أن يعامل بعضهم بعضا بروح الإخاء.

Yūladu ğamīʿu ’n-nāsi ʾaḥrāran mutasāwīna fī ’l-karāmati wa-’l-ḥuqūqi. Wa-qad wuhibū ʿaqlan wa-ḍamīran wa-ʿalayhim ʾan yuʿāmila baʿḍuhum baʿḍan bi-rūḥi ’l-ʾiẖāʾi.

```
\cap yUladu ^gamI‘u an-nAsi ’a.hrAraN mutasAwIna fI
     al-karAmaTi wa-al-.huqUqi.
\cap wa-qad wuhibUA ‘aqlaN wa-.damIraN wa-‘alayhim ’an
     yu‘Amila ba‘.duhum ba‘.daN bi-rU.hi al-’i_hA’i.
```

Encode Arabic

biruwHi {loIixaA'i ← بِرُوحِ آلْإِخَاءِ ← bi-rU.hi al-'i_hA'i

Implemented in Perl and available on CPAN as Encode-Arabic:

```
$encoded = encode "buckwalter", decode "arabtex", $decoded
$encoded = encode("buckwalter", decode("arabtex", $decoded))
```

Implemented in Haskell and available along with ElixirFM:

```
encoded = encode Buckwalter $ decode ArabTeX decoded
encoded = encode Buckwalter (decode ArabTeX decoded)
encoded = (encode Buckwalter . decode ArabTeX) decoded
```

```
[cmd] decode ArabTeX < decode.d | encode Buckwalter > encode.d
```

Encode Arabic

biruwHi {loIixaA'i ← بِرُوحِ آلْإِخَاءِ ← bi-rU.hi al-'i_hA'i

Implemented in Perl and available on CPAN as Encode-Arabic:

```
$encoded = encode "buckwalter", decode "arabtex", $decoded
$encoded = encode("buckwalter", decode("arabtex", $decoded))
```

Implemented in Haskell and available along with ElixirFM:

```
encoded = encode Buckwalter $ decode ArabTeX decoded
encoded = encode Buckwalter (decode ArabTeX decoded)
encoded = (encode Buckwalter . decode ArabTeX) decoded
```

```
[cmd] decode ArabTeX < decode.d | encode Buckwalter > encode.d
```

Encode Arabic

biruwHi {loIixaA'i ← بِرُوحِ آلْإِخَاءِ ← bi-rU.hi al-'i_hA'i

Implemented in Perl and available on CPAN as Encode-Arabic:

```
$encoded = encode "buckwalter", decode "arabtex", $decoded
$encoded = encode("buckwalter", decode("arabtex", $decoded))
```

Implemented in Haskell and available along with ElixirFM:

```
encoded = encode Buckwalter $ decode ArabTeX decoded
encoded = encode Buckwalter (decode ArabTeX decoded)
encoded = (encode Buckwalter . decode ArabTeX) decoded
```

```
[cmd] decode ArabTeX < decode.d | encode Buckwalter > encode.d
```

Encode Arabic

biruwHi {loIixaA'i ← بِرُوحِ آلْإِخَاءِ ← bi-rU.hi al-'i_hA'i

Implemented in Perl and available on CPAN as Encode-Arabic:

```
$encoded = encode "buckwalter", decode "arabtex", $decoded
$encoded = encode("buckwalter", decode("arabtex", $decoded))
```

Implemented in Haskell and available along with ElixirFM:

```
encoded = encode Buckwalter $ decode ArabTeX decoded
encoded = encode Buckwalter (decode ArabTeX decoded)
encoded = (encode Buckwalter . decode ArabTeX) decoded
```

```
[cmd] decode ArabTeX < decode.d | encode Buckwalter > encode.d
```

Encode Arabic

biruwHi {loIixaA'i ← بِرُوحِ آلْإِخَاءِ ← bi-rU.hi al-'i_hA'i

Implemented in Perl and available on CPAN as Encode-Arabic:

```
$encoded = encode "buckwalter", decode "arabtex", $decoded
$encoded = encode("buckwalter", decode("arabtex", $decoded))
```

Implemented in Haskell and available along with ElixirFM:

```
encoded = encode Buckwalter $ decode ArabTeX decoded
encoded = encode Buckwalter (decode ArabTeX decoded)
encoded = (encode Buckwalter . decode ArabTeX) decoded
```

```
[cmd] decode ArabTeX < decode.d | encode Buckwalter > encode.d
```

Morphology Disambiguation

Arabic is a language of rich morphology, both derivational and inflectional, with highly ambiguous orthography.

Morphology Disambiguation

Arabic is a language of rich morphology, both derivational and inflectional, with highly ambiguous orthography.

Boundaries of syntactic units, tokens, are obscure in writing—orthographical words, strings, consist of up to four lexemes.

Morphology Disambiguation

Arabic is a language of rich morphology, both derivational and inflectional, with highly ambiguous orthography.

Boundaries of syntactic units, tokens, are obscure in writing—orthographical words, strings, consist of up to four lexemes.

Disambiguation encompasses subproblems like tokenization

Morphology Disambiguation

Arabic is a language of rich morphology, both derivational and inflectional, with highly ambiguous orthography.

Boundaries of syntactic units, tokens, are obscure in writing—orthographical words, strings, consist of up to four lexemes.

Disambiguation encompasses subproblems like tokenization, full morphological tagging or its simplified 'part-of-speech' versions

Morphology Disambiguation

Arabic is a language of rich morphology, both derivational and inflectional, with highly ambiguous orthography.

Boundaries of syntactic units, tokens, are obscure in writing—orthographical words, strings, consist of up to four lexemes.

Disambiguation encompasses subproblems like tokenization, full morphological tagging or its simplified 'part-of-speech' versions, lemmatization

Morphology Disambiguation

Arabic is a language of rich morphology, both derivational and inflectional, with highly ambiguous orthography.

Boundaries of syntactic units, tokens, are obscure in writing—orthographical words, strings, consist of up to four lexemes.

Disambiguation encompasses subproblems like tokenization, full morphological tagging or its simplified 'part-of-speech' versions, lemmatization, diacritization or restoration of the structural components of words

Morphology Disambiguation

Arabic is a language of rich morphology, both derivational and inflectional, with highly ambiguous orthography.

Boundaries of syntactic units, tokens, are obscure in writing—orthographical words, strings, consist of up to four lexemes.

Disambiguation encompasses subproblems like tokenization, full morphological tagging or its simplified 'part-of-speech' versions, lemmatization, diacritization or restoration of the structural components of words, plus combinations thereof.

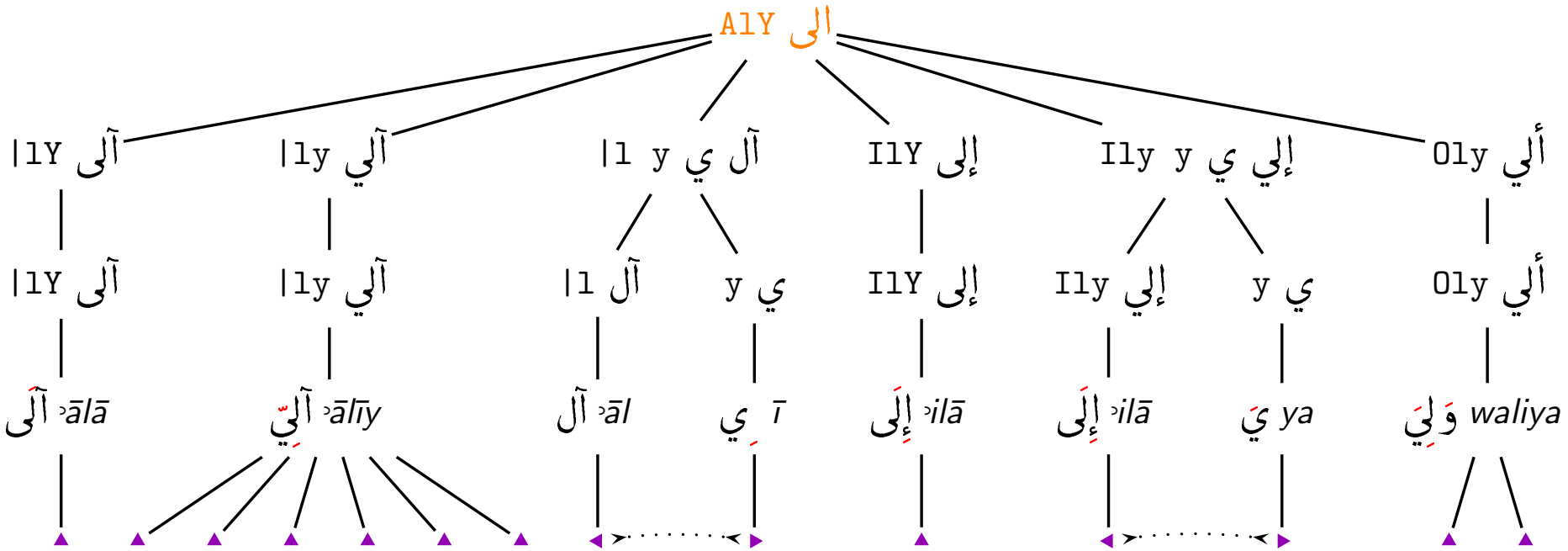
Linear Lists

Suppose you can list morphological analyses for a given input string ...

Morphs	Form	Token Tag	Lemma	Glosses per Morph
|laY+(null)	ʾālā	VP-A-3MS--	ʾālā	promise/take an oath + he/it
|liy˜	ʾālīy	A---------	ʾālīy	mechanical/automatic
|liy˜+u	ʾālīy-u	A-------1R	ʾālīy	mechanical ... + [def.nom.]
|liy˜+i	ʾālīy-i	A-------2R	ʾālīy	mechanical ... + [def.gen.]
|liy˜+a	ʾālīy-a	A-------4R	ʾālīy	mechanical ... + [def.acc.]
|liy˜+N	ʾālīy-un	A-------1I	ʾālīy	mechanical ... + [indef.nom.]
|liy˜+K	ʾālīy-in	A-------2I	ʾālīy	mechanical ... + [indef.gen.]
|l +	ʾāl	N--------R	ʾāl	family/clan +
+ iy	-ī	S----1-S2-	ī	+ my
IilaY	ʾilā	P---------	ʾilā	to/towards
Iilay +	ʾilay	P---------	ʾilā	to/towards +
+ ya	-ya	S----1-S2-	ya	+ me
Oa+liy+(null)	ʾa-lī	VIIA-1-S--	waliya	I + follow/come after + [ind.]
Oa+liy+a	ʾa-liy-a	VISA-1-S--	waliya	I + follow/come after + [sub.]

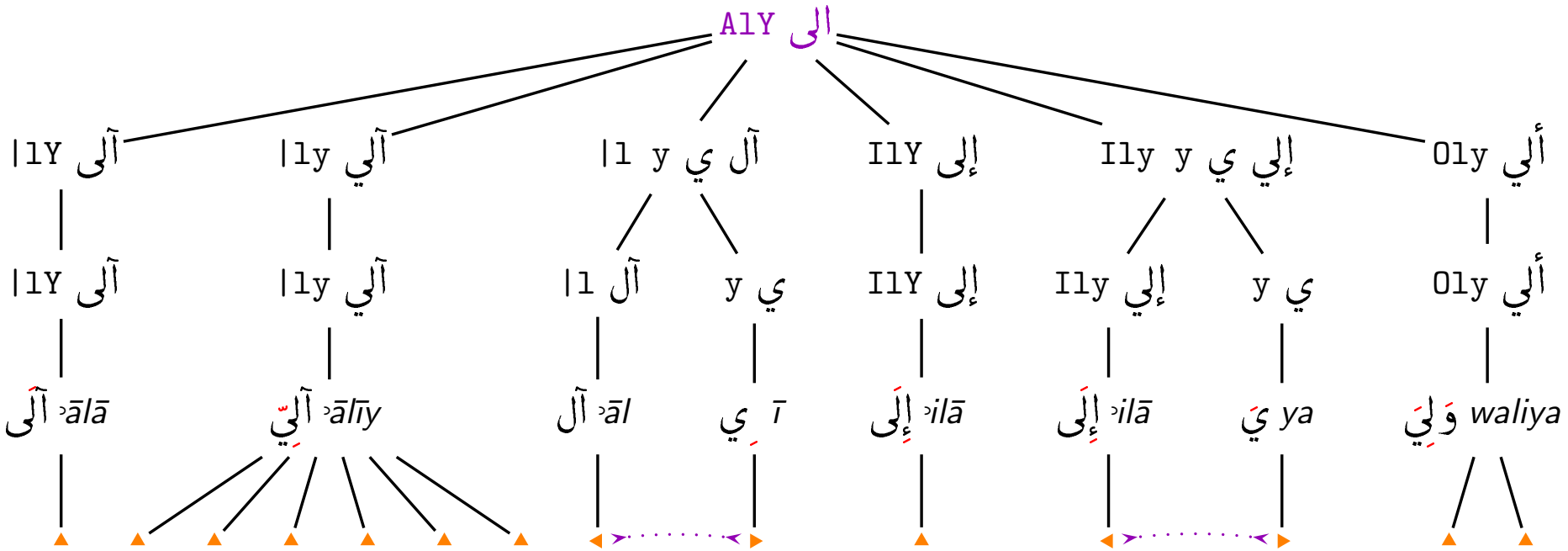
MorphoTrees

. . . organize the analyses into a hierarchy with the string as its root



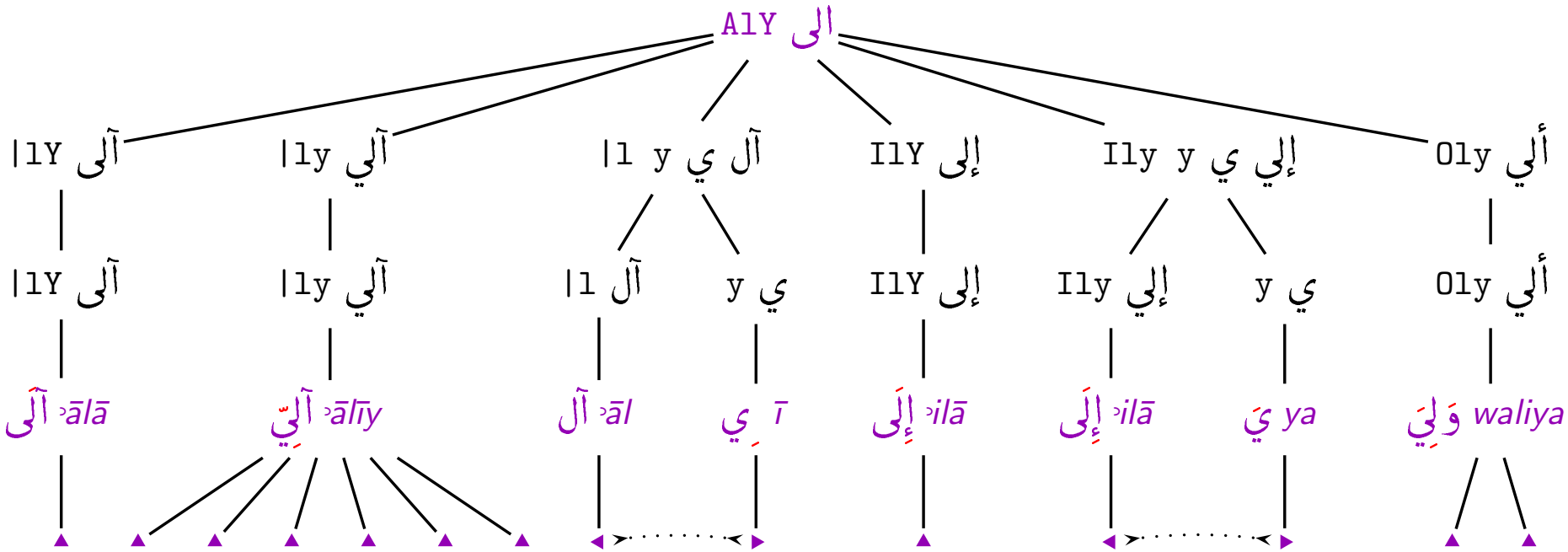
MorphoTrees

. . . organize the analyses into a hierarchy with the string as its root and the full tokens as the leaves



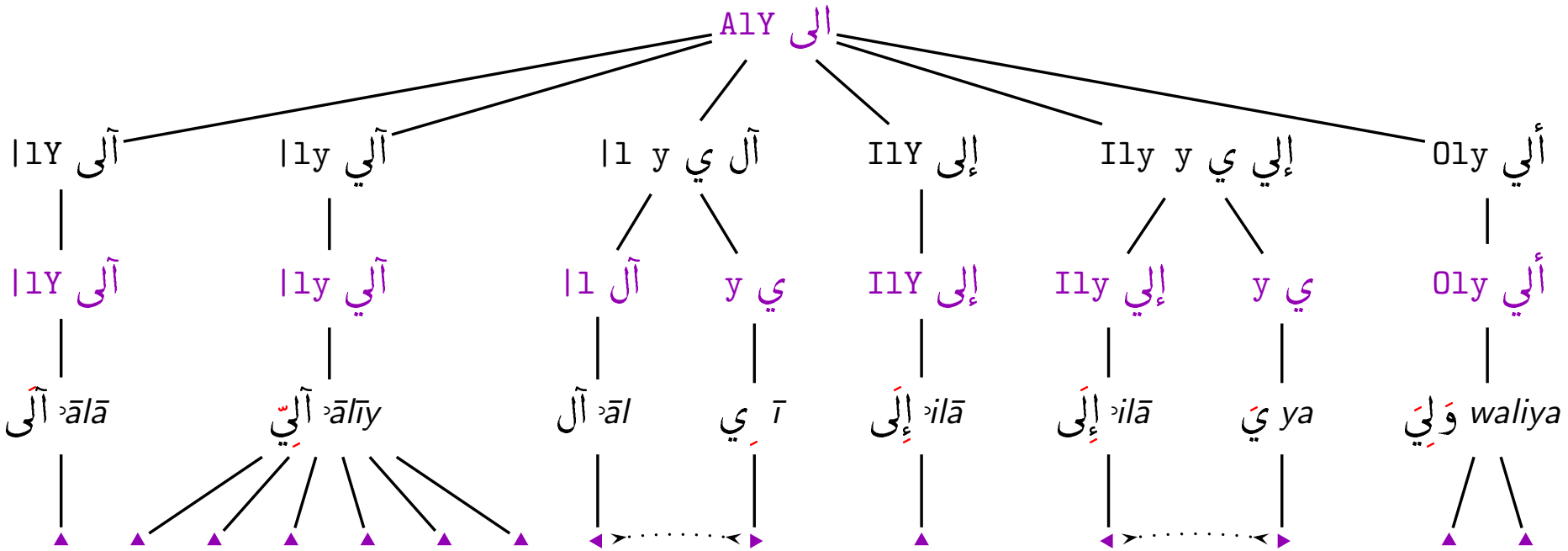
MorphoTrees

. . . organize the analyses into a hierarchy with the string as its root and the full tokens as the leaves, grouped by their lemmas



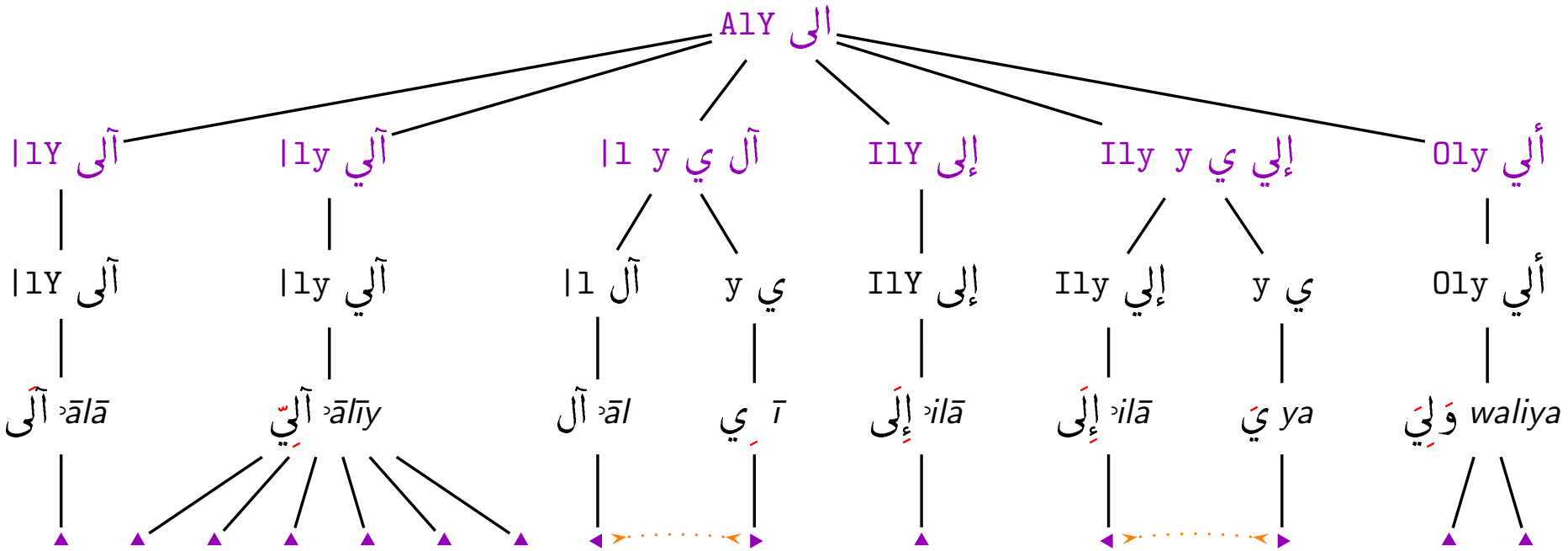
MorphoTrees

. . . organize the analyses into a hierarchy with the string as its root and the full tokens as the leaves, grouped by their lemmas, canonical forms

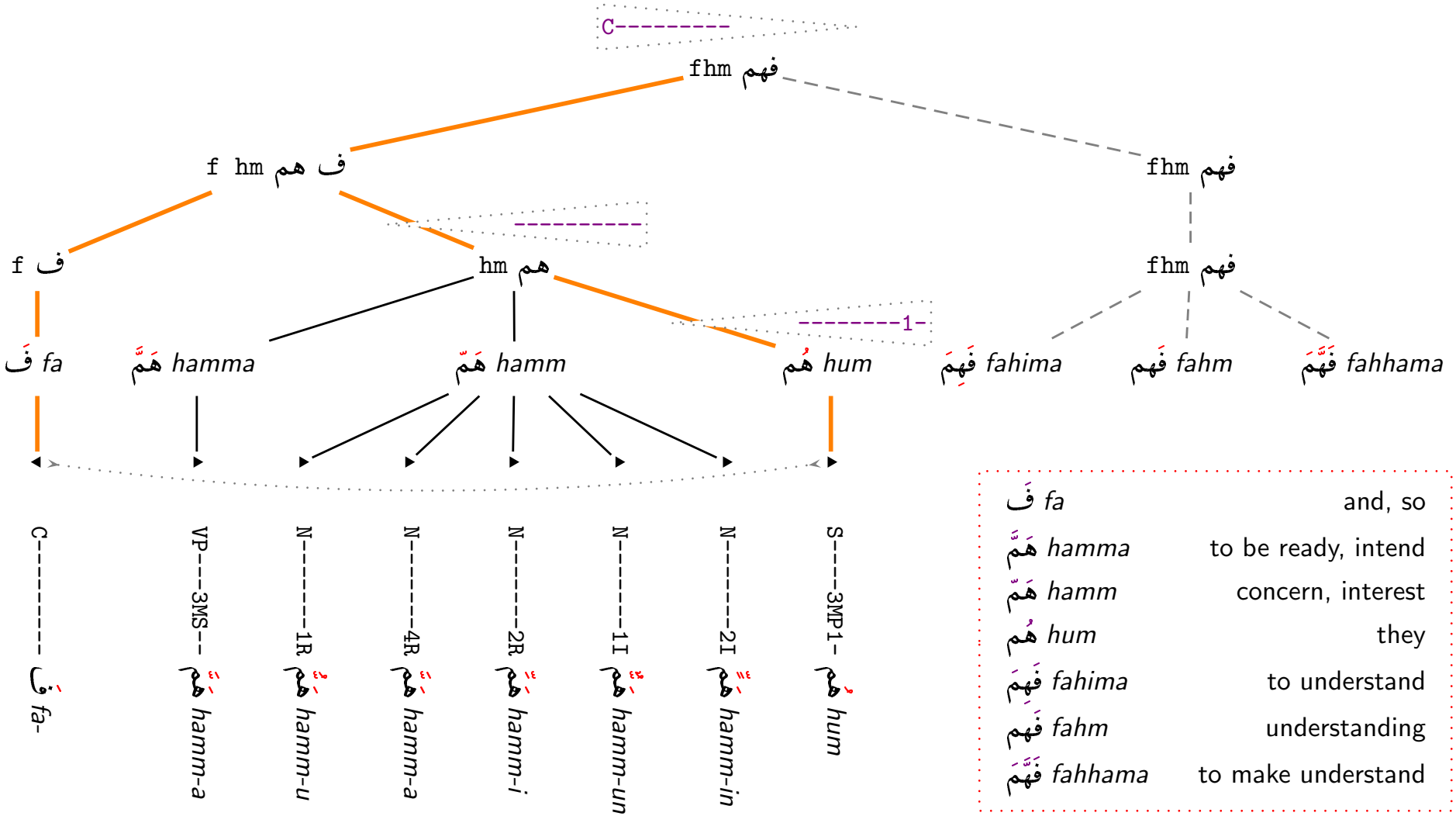


MorphoTrees

. . . organize the analyses into a hierarchy with the string as its root and the full tokens as the leaves, grouped by their lemmas, canonical forms and partitionings of the string into such forms:



Multi-Modal Annotation



فَ *fa*	and, so
هَمَّ *hamma*	to be ready, intend
هَمّ *hamm*	concern, interest
هُم *hum*	they
فَهِمَ *fahima*	to understand
فَهم *fahm*	understanding
فَهَّمَ *fahhama*	to make understand

Outline

1. Introduction
2. Morphological Theory
   - Incremental vs. Realizational
   - Lexical vs. Inferential
   - Functional Arabic Morphology
3. Implementation Design
   - ElixirFM
   - Paradigms, parameters, ...
   - Elixir Lexicon
   - FM Generic
4. Extensions
   - Encode Arabic
   - MorphoTrees
5. References

- Buckwalter, Tim. Buckwalter Arabic Morphological Analyzer 1.0. LDC catalog number LDC2002L49, ISBN 1-58563-257-0. 2002
- Forsberg, Markus and Aarne Ranta. Functional Morphology. Proceedings of ICFP 2004, pages 213–223. ACM Press. 2004
- Lagally, Klaus. ArabTeX: Typesetting Arabic and Hebrew, User Manual Version 4.00. Technical Report 2004/03, Fakultät Informatik, Universität Stuttgart. 2004
- Ljunglöf, Peter. Pure Functional Parsing. An Advanced Tutorial. Licenciate thesis, Göteborg University & Chalmers University of Technology. 2002.
- Smrž, Otakar and Petr Pajas. MorphoTrees of Arabic and Their Annotation in the TrEd Environment. Proceedings of the NEMLAR Conference 2004, pages 38–41. 2004
- Stump, Gregory T. Inflectional Morphology. A Theory of Paradigm Structure. Cambridge Studies in Linguistics. Cambridge University Press. 2001