



BASIS
TECHNOLOGY



Tailoring UAX #29 Word Breaking for Arabic Text from the Wild

Thomas Emerson
Software Architect

28th Internationalization & Unicode Conference
Orlando, Florida, USA

Basis Technology Corporation

P 617.386.2000
800.697.2062 (toll-free)
F 617.386.2020
W info@basistech.com
www.basistech.com



Overview

- What is UAX #29?
 - Types of breaks
 - Why do we need it?
 - Tailoring?
- Arabic Orthography
 - The Reality of the Web
- Tailoring for Arabic
- Implementation Issues



What is UAX #29?

- Unicode Standard Annex
 - Integral part of The Unicode Standard...
 - ...Published as a separate document
 - UAX #29 is *informative*



What is UAX #29?

- “Text Boundaries”
 - Defines *general guidelines* for finding *default* boundaries between entities
 - *Grapheme Clusters*
 - *Words*
 - *Sentences*
 - Line breaking is handled by UAX #14



What is UAX #29?

- Grapheme Clusters
 - “Character” from a user’s perspective
 - Visual representation different from the underlying representation
 - *Combining character sequences*
 - ü = <0075, 0308>
 - *Composed han’gul syllables*
 - 가ㅏ = <1100, 1161>



What is UAX #29?

- Sentence Boundaries
 - Different from line breaking
 - *Certain characters can prohibit line breaks*
 - Generic rules get you only so far
 - Difficult (for some languages) without linguistic analysis
 - *“Mr. Trump said, ‘You’re fired!’”*
 - Easy for others
 - 胡主席此次访加正值中加建交35周年。



What is UAX #29?

- Word Breaking
 - Why do it at all?
 - *Double-click Selection*
 - *“Full word” search*
 - *Text indexing*
 - *Linguistic analysis*
 - It’s hard to do in a general way cross-linguistically



What is UAX #29?

- Word Breaking
 - words = re.split("\s+", l)
 - *This is too simplistic*
 - 你會說中文嗎?
 - *Where are the words here? Not after each character!*
 - 私はアメリカ人である。
 - *This contains at least three words*



What is UAX #29?

- General Guidelines for Default Boundaries
 - Define where a break can occur...
 - ...not the entities themselves...
 - ...based on character property values
- Implies that:
 - Fast and (hopefully compact) representation of the properties
 - Fast implementation of the rules in an FSM
 - *Other implementations possible too*



What is UAX #29?

- Rules and properties fully described in UAX #29
 - No break within a GC
 - No break between letters
 - No break between letters and certain punctuation
 - No break between numbers and letters
 - No break between numbers and certain punctuation
 - No break between katakana



What is UAX #29?

- Defaults can be “tailored”
 - Adapted for specific purposes or languages
 - For example, “.” is not allowed between letters, prohibiting “U.S.A”



What is UAX #29?

- Tailoring Techniques
 - Data only: redefine the assignment of properties to code-points
 - Code only: specific implementation differences
 - Hybrid: Code/rule changes with specific property values



Arabic Orthography

- Arabic written right-to-left
 - Numbers written left-to-right
- Space separated words, but...
 - ... enclitic pronouns
 - ... prepositions, conjunctions, definite article are joined
 - والكتاب
 - “*and the book*”



Arabic in the Wild

- Normalization is absolutely required when processing Arabic on the Web
 - `الإعصاركاتريناالذيضرب`
 - Presentation forms show up, even in HTML.



Arabic in the Wild

- Still have problems...
 - Use of ر as a number separator
 - 310,70
 - Joining words after ة
 - الحقيقة يجعل
 - Use of 0 as the full stop
 - دورة النوايا000
 - *Arabic-Indic Number 0 is .*



Arabic in the Wild

- And more...
 - Hamza and madda on alef variation (أ آ إ)
 - Word final *ya* with ي or ی
 - ‘anna أ vs. ‘inna إ
 - Concatenated morphemes
 - *Function words regularly joined*
 - لابد ، مايرام
 - *Names using ‘abd, عبد, regularly written with or without following space*



Arabic in the Wild

- And more...
 - Visual encoding instead of logical
 - This happens often in PDF documents
 - *Presentation forms in visual ordering*



Tailoring for Wild Arabic

- For what purpose?
 - Linguistic analysis
 - Orthographic words to feed to morphological analysis
 - Cannot account for all orthographic variation that is observed through data changes alone



Implementation Issues

- UAX #29 defaults can handle use of `؁` within a number
- Add a rule to handle token embedded *tehamarbuta*.
 - Gets its own boundary property
 - Gets its own rule: break always afterwards



Implementation Issues

- Representation of boundary properties
 - Lookup table mapping code point to property
 - “Compact Array” is compact and fast
 - *10 KB for Unicode 4.1*
- Representing the Rules
 - Logic-in-code
 - State Machines

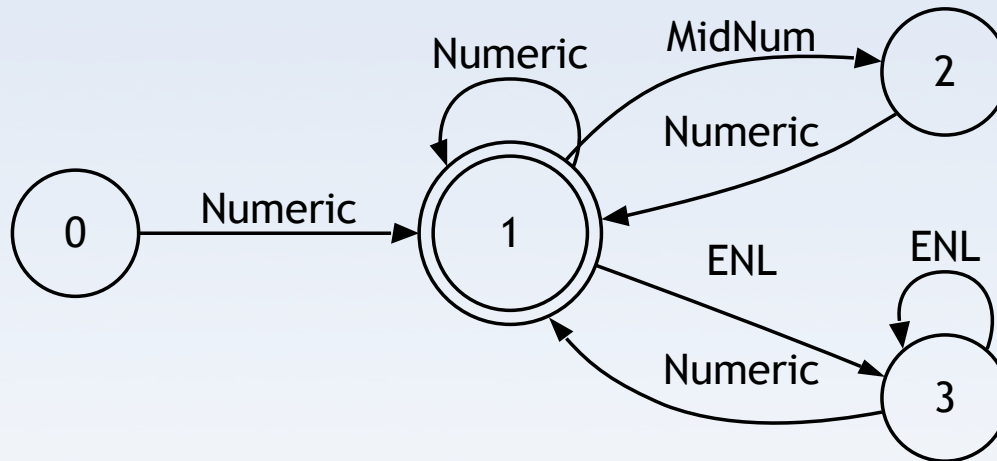


Implementation Issues

- Generating and maintaining FSMs by hand is error-prone and annoying
 - ICU compiler for RBBI
 - Writing and debugging tools time consuming
- Doing it by hand
 - Generate the regular expressions
 - *Always look for the longest match*
 - “Compile” these to a state table

Implementation Issues

- `Numeric+ ((MidNum | ExtendedNumLet+) Numeric+)*`





Implementation Issues

- Handling other variation requires integration of the morphological analyzer with the tokenizer!
 - Tokenization drives morphological variation generation
 - Modify engine UAX #29 to “detect” presence of Arabic and special case analysis in these situations
- If you are interested in Arabic MA, see me later.

شكرا!



<http://www.basistech.com/knowledge-center/>