R in linguistic studies

SPbR #17 / Applied R Munich Christmas Edition

Anton Antonov

@tonytonov

2020-12-11

dxFeed

How it started

Alexander Piperski, Senior Lecturer, Research Fellow at HSE

- https://www.hse.ru/en/org/persons/34802244
- Constructing Languages: From Esperanto to Dothraki

Anton Somin, Visiting Scholar, Research Fellow at HSE

- https://www.hse.ru/en/org/persons/34802260

How it started

Alexander Piperski, Senior Lecturer, Research Fellow at HSE

- https://www.hse.ru/en/org/persons/34802244
- Constructing Languages: From Esperanto to Dothraki

Anton Somin, Visiting Scholar, Research Fellow at HSE

- https://www.hse.ru/en/org/persons/34802260

Linguistics: phonetics and graphics (in Russian)
https://edu.sirius.online/#/course/88

Course overview

Linguistics: phonetics and graphics

13 modules, e.g. *Vowels* or *Letter-based writing systems*

25 video lectures, 333 exercises

Course overview

Linguistics: phonetics and graphics

13 modules, e.g. *Vowels* or *Letter-based writing systems*

25 video lectures, 333 exercises

Enrolled: 13894, passed: 664

Course overview

Linguistics: phonetics and graphics

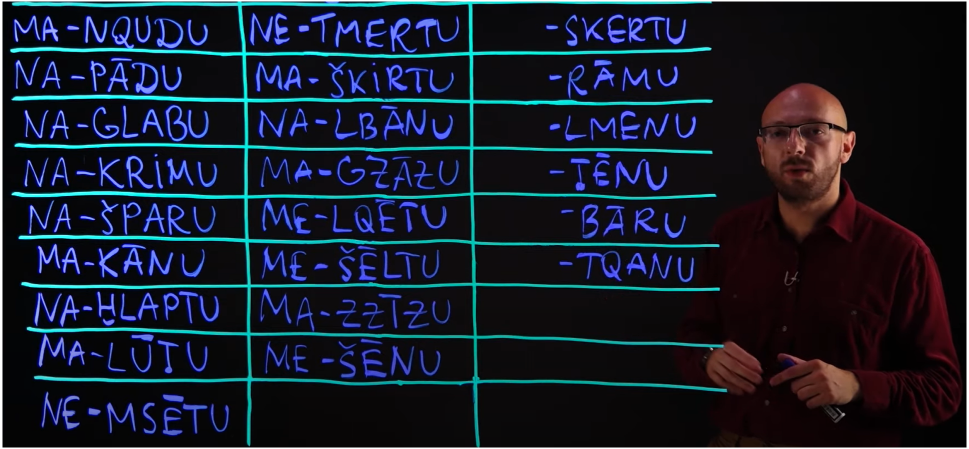
13 modules, e.g. *Vowels* or *Letter-based writing systems*

25 video lectures, 333 exercises

Enrolled: 13894, passed: 664

```
R> sprintf("%1.2f%%", 100 * 664 / 13894)
[1] "4.78%"
```

Typical exercise



MA-NQUDU	NE-TMERTU	-SKERTU
NA-PĀDU	MA-ŠKiRTU	-RĀMU
NA-GLABU	NA-LBĀNU	-LMENU
NA-KRiMU	MA-GZĀZU	-ṬĒNU
NA-ŠPARU	ME-LQĒTU	-BĀRU
MA-KĀNU	ME-ŠĒLTU	-TQANU
NA-ḪLAPTU	MA-ZZĪZU	
MA-LŪṬU	ME-ŠĒNU	
NE-MSĒTU		

Figure 1: Akkadian language, Mesopotamia, 3rd–1st millenia BC

That one

In Tibetian, there are four tones: flat–high (I), rising (II), rising–falling (III) and falling (IV).

dku	side	I	ston	autumn	I	mi	man	?
kun	whole	I	do	pair	II	su	who	?
mgo	head	II	sde	tribe	II	bur	bolt	?
nub	west	III	lkog	mystery	IV	mdog	colour	?
sad	frost	IV	bod	Tibet	III	lug	sheep	?
re	each	II	pag	brick	IV	kug	hook	?
til	sesame	I	zil	glare	II	ldum	vegetables	?
gzig	leopard	III	ngang	character	II			

Table 1: Go figure

What to do?

Try really hard

- Two words matched with 80% certainty

What to do?

Try really hard

- Two words matched with 80% certainty

Google it

- Doesn’t help: too hardcore

What to do?

Try really hard

- Two words matched with 80% certainty

Google it

- Doesn't help: too hardcore

Learn Tibetian

- Found one word!

What to do?

Try really hard

- Two words matched with 80% certainty

Google it

- Doesn't help: too hardcore

Learn Tibetian

- Found one word!

Brute-force

- By hand?
- Improvise

Brute-force complexity

- Number of combinations of assigning n words into k categories:

$$k \times k \times \cdots \times k = k^n$$

Brute–force complexity

- Number of combinations of assigning n words into k categories:

$$k \times k \times \cdots \times k = k^n$$

- Same, but with at least one word in each category: $k! s_k^{II}(n)$, where $s_k^{II}(n)$ are *Stirling Numbers of the second kind*

$$s_k^{II}(n) = k s_k^{II}(n-1) + s_{k-1}^{II}(n-1),$$

$$s_0^{II}(0) = 1, s_0^{II}(n) = s_n^{II}(0) = 0$$

Brute–force complexity

- Number of combinations of assigning n words into k categories:

$$k \times k \times \cdots \times k = k^n$$

- Same, but with at least one word in each category: $k!s_k^{II}(n)$, where $s_k^{II}(n)$ are *Stirling Numbers of the second kind*

$$s_k^{II}(n) = ks_k^{II}(n-1) + s_{k-1}^{II}(n-1),$$

$$s_0^{II}(0) = 1, s_0^{II}(n) = s_n^{II}(0) = 0$$

- Reference: M. Axenovich, T. Ueckerdt, J. Rollin and S. Walzer (2017). *Lecture Notes. Combinatorics.*

Quick counts

```
R> n <- 7; k <- 4; k^n
[1] 16384
```

Quick counts

```
R> n <- 7; k <- 4; k^n
[1] 16384
```

R has everything!

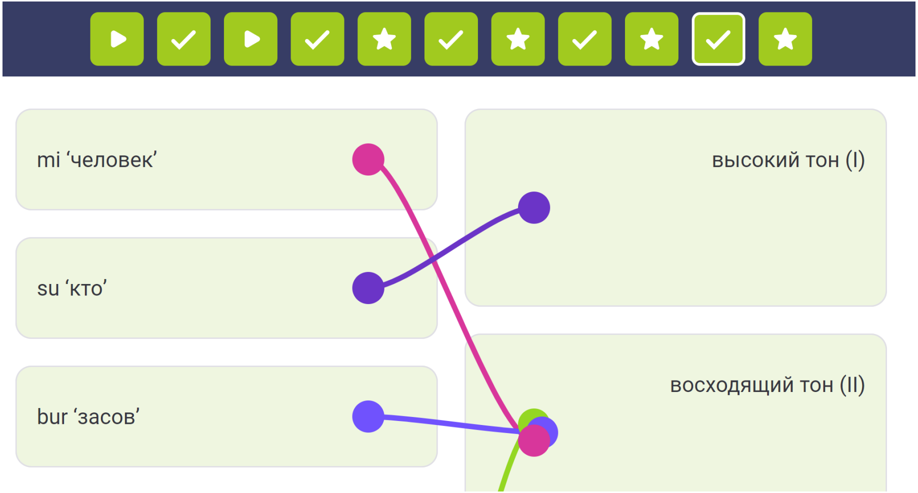
Quick counts

```
R> n <- 7; k <- 4; k^n
[1] 16384
```

R has everything!

```
R> pipe <- function(n, k) {
       gmp::Stirling2(n, k) * gmp::factorialZ(k)
   }
R> pipe(n, k)
Big Integer ('bigz') :
[1] 8400
```

Plus the UI



mi ‘человек’

su ‘кто’

bur ‘засов’

высокий тон (I)

восходящий тон (II)

Figure 2: UI form of the exercise

Improvisation plan

https://stepik.org/course/497 (in Russian)

Improvisation plan

https://stepik.org/course/497 (in Russian)

Action plan:

- Find out if there’s an API for submissions

Improvisation plan

https://stepik.org/course/497 (in Russian)

Action plan:

- Find out if there's an API for submissions
- Study the API request

Improvisation plan

https://stepik.org/course/497 (in Russian)

Action plan:

- Find out if there's an API for submissions
- Study the API request
- Mimic the button press in R

Improvisation plan

https://stepik.org/course/497 (in Russian)

Action plan:

- Find out if there’s an API for submissions
- Study the API request
- Mimic the button press in R
- Prepare the code that implements brute-force

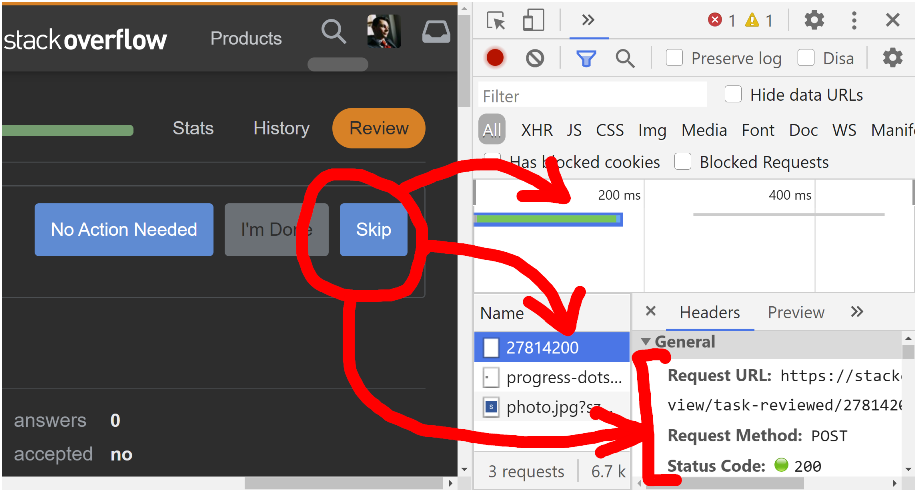
Improvisation plan

https://stepik.org/course/497 (in Russian)

Action plan:

- Find out if there's an API for submissions
- Study the API request
- Mimic the button press in R
- Prepare the code that implements brute-force
- Sit back and relax

Find API



Request URL: https://stack
view/task-reviewed/278142

Request Method: POST

Status Code: 200

Figure 3: Chrome “Developer tools” (Win F12)

Study request

- *General:*
  - Request URL: `https://edu.sirius.online/xxxxxx/solve`
  - Request Method: `POST`
  - Status Code: `200 OK`

Study request

- *General:*
  - Request URL: `https://edu.sirius.online/xxxxxx/solve`
  - Request Method: `POST`
  - Status Code: `200 OK`
- *Request Headers:*
  - Authorization: `Bearer eyJh[...]`
  - Accept: `application/json`
  - Content-Length: `59`

Study request

- *General:*
  - Request URL: `https://edu.sirius.online/xxxxxx/solve`
  - Request Method: `POST`
  - Status Code: `200 OK`
- *Request Headers:*
  - Authorization: `Bearer eyJh[...]`
  - Accept: `application/json`
  - Content-Length: `59`
- *Request Payload:*
  - `{"solution":[[7,9],[6,11],[2,8],[5,10],[1,8],[3,9],[4,11]]}`

Study request

- *General:*
  - Request URL: https://edu.sirius.online/xxxxxx/solve
  - Request Method: POST
  - Status Code: 200 OK
- *Request Headers:*
  - Authorization: Bearer eyJh[...]
  - Accept: application/json
  - Content-Length: 59
- *Request Payload:*
  - {"solution":[[7,9],[6,11],[2,8],[5,10],[1,8],[3,9],[4,11]]}
- *Response:*
  - {"verdict":"wrong"}

Identify payload rules

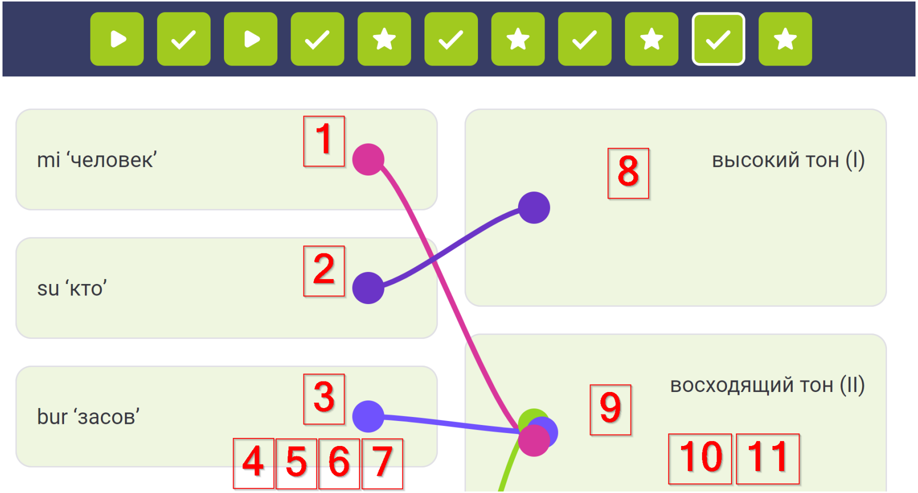


Figure 4: Number pairs correspond to user choices

Mimic button press

```
R> v <- c(8, 8, 9, 9, 10, 11, 11)
R> solution <- jsonlite::toJSON(cbind(1:7, v))
R> body <- glue::glue('{"solution":<solution>}',
                      .open = "<", .close = ">")
R> body
{"solution":[[1,8],[2,8],[3,9],[4,9],[5,10],[6,11],[7,11]]}
R> nchar(body)
[1] 59
```

Mimic button press

```
R> v <- c(8, 8, 9, 9, 10, 11, 11)
R> solution <- jsonlite::toJSON(cbind(1:7, v))
R> body <- glue::glue('{"solution":<solution>}',
                      .open = "<", .close = ">")
R> body
{"solution":[[1,8],[2,8],[3,9],[4,9],[5,10],[6,11],[7,11]]}
R> nchar(body)
[1] 59
```

Try `POST`ing this request, we want response `200 OK`

Obligatory disclaimers

- In other circumstances, this may violate ToS/EULA
- The API load should be reduced to a minimum
- The API backend may take action (halt, ban)
- The course is over and the backend has changed since then

Implement brute-force

Base R is incredibly rich in low-level abstractions!

Implement brute-force

Base R is incredibly rich in low-level abstractions!

```
R> expand.grid(1:2, c("A", "B"))
  Var1 Var2
1    1    A
2    2    A
3    1    B
4    2    B
```

Implement brute-force

```
R> index_category <- 8:11
R> all_combn <- do.call(what = expand.grid, args = replicate(
                                  n = length(index_category),
                                  expr = index_category,
                                  simplify = F))
R> head(all_combn)
  Var1 Var2 Var3 Var4
1    8    8    8    8
2    9    8    8    8
3   10    8    8    8
4   11    8    8    8
5    8    9    8    8
6    9    9    8    8
```

Prepare the code

```
R> library(httr)
R> p <- POST('https://edu.sirius.online/xxxxxx/solve',
                 body = body,
                 add_headers(.headers = c(
                     'Accept' = 'application/json',
                     [...]
                     'Content-Length' = nchar(body),
                     'Content-Type' = 'application/json',
                     [...])))
R> if (content(p)$verdict != "wrong") {print(solution); stop("Done!")}
```

Sit back and relax

- Don’t forget `Sys.sleep(1)`

Sit back and relax

- Don’t forget `Sys.sleep(1)`
- Skip combination if not all categories (8–11) are present

Sit back and relax

- Don’t forget `Sys.sleep(1)`
- Skip combination if not all categories (8–11) are present
- Correct answer found in 2 minutes

Sit back and relax

- Don't forget `Sys.sleep(1)`
- Skip combination if not all categories (8–11) are present
- Correct answer found in 2 minutes
- Let me know if you know how to solve it!

Sit back and relax

- Don’t forget `Sys.sleep(1)`
- Skip combination if not all categories (8–11) are present
- Correct answer found in 2 minutes
- Let me know if you know how to solve it!

Thanks!

https://github.com/tonytonov/talks

`www.linkedin.com/in/tonytonov`