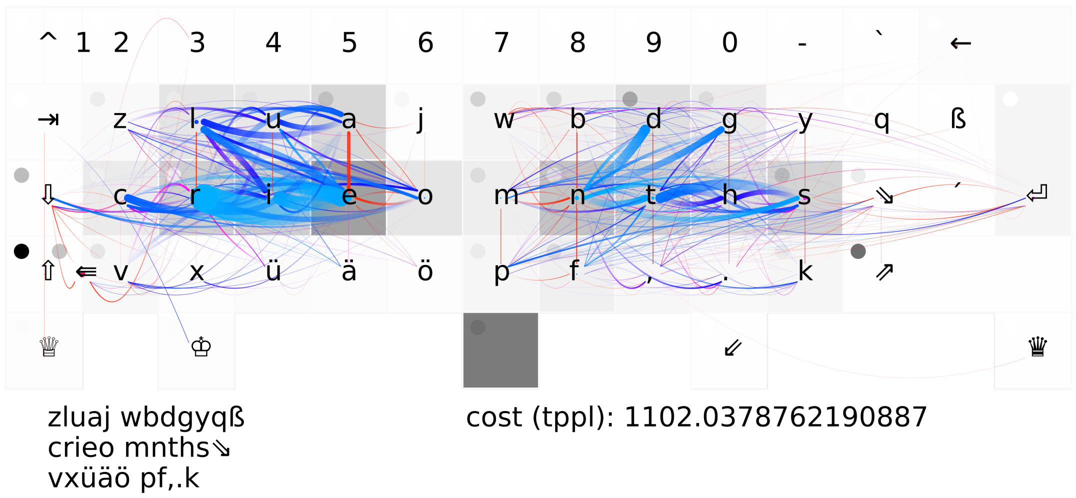
Evolutionary optimization of keyboard layouts

Whatever the future brings, I will be typing a lot. Best make sure I use good tools.


^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ z l u a j w b d g y q ß
⇩ c r i e o m n t h s ⇘ ´ ⏎
⇧ ⇚ v x ü ä ö p f , . k ⇗
♕ ♔ ⇙ ♛
zluaj wbdgyqß
crieo mnths⇘
vxüäö pf,.k
cost (tppl): 1102.0378762190887

Update 2023-08: Dario Götz found the noted layout with his optimizer, a layout that looks pretty good: Noted (there's also an English version of the text).

Update 2023: There's nowadays a much faster optimizer by Dario Götz using the same concept of typing cost calculation. It enables much nimbler experimentation: `https://github.com/dariogoetz/keyboard_layout_optimizer`

Update 2022-11: Since spring 2021 mine has the z in the upper right corner. The current version is described on neo-layout.org/Layouts/mine/.

Contents

History

It was around 2011 and I worked on my Diploma thesis as I found the community of people around the Neo Layout, German Keyboard-Nerds working on better keyboard layouts than the accident of history[1] that is QWERTY (QWERTZ in Germany).

At a similar time, another typing-enthusiast joined, and he was unhappy with the Neo-Layout, since he was used to typing English with Dvorak and hoped that Neo would provide a typing experience like Dvorak, but for German text (where Dvorak fails badly). He started to write a program to improve the layout so it would feel more like Dvorak, but work well for German. I was fascinated, but found his approach lacking and thought "I can do that better".

Over the next year we both improved the systems. Competition arose. He later adopted an existing optimizer from the Neo-Community along with the `AdNW` layout and I kept improving my optimizer with more and more criteria we found in the Neo-Community by conducting collaborative typing tests to see where layouts were still lacking.

You can get my optimizer from hg.sr.ht/~arnebab/evolve-keyboard-layout.

Then I found a layout that looked very good, but for which I lacked experience with high speed typing — things that hurt when typing fast again — and I decided to do a 3 year test to see which problems would arise in high-speed typing. In the meantime one who called themself Effchen used a simplified version of my optimizer and combined that with lots of manual improvements and intuition to create the Bone-Layout.

My 3 year test extended to 7 years while I struggled to find the time for implementing fixes for problems I found while typing fast, but in January 2020 I used an Aarlaubsday at work to finish the regularity calculations — the last missing piece for better layouts — and around the middle of 2020 I finally had a new layout to test.

It is called mine, and you can find functional layout descriptions of it from the Neo-Community:

→ **Mine** ←

[1]The story how QWERTY became dominant is surrounded by fiction — and even the article in the previous link builds on earlier work that — while unveiling problems in reasoning of others — succumbs to its own pitfalls to argument against path dependence, because it argues that too little retraining benefit in raw typing speed to be economically viable for companies to retrain all typists does not show a market failure or path dependence, even though it shows exactly both, because its argumentation would not apply if (a) typing speed were mostly dependent on training time (which it is, so the old layout always had an advantage in any competition), or (b) people learned an optimized layout as *first* keyboard layout. The second option was not even considered. What that article did not have to check its own claims is something we now have: actual metrics of layouts based on identified problematic movements. That's what you can see in the graphics on this page: QWERTY is bad for typing.

Yesterday I began to type with it. It will **still need some tweaks**, because no optimizer can capture the whole complexity of the typing process, but from now on you are invited to join.

Required changes: Already known weaknesses of the mine-version above which *must be fixed* before a final version:

- ~~*mpf* is hard to type, because it is all on the index finger. This is a not so common but too common triple in German.~~ since we stopped talking about Impfungen every day, this stopped being annoying.

Here I want to share some resources with background information.

Typing cost calculation

My optimizer calculates a score for each layout that gives the cost of typing: The **total penalty per letter**. That score is an aggregate from many different criteria. The most important are:

key position how easy are keys to reach, weighted by how common they are in writing.

finger repeats how often do you have to use the same finger twice

top <-> bottom how often do you have to move with the same finger between the top and the bottom row

(rows²/dist)² how much do you have to move vertically? This minimizes stretch of the fingers.

handswitching how often do you switch hands (long sequences on the same hand get complex)

same hand after unbalancing if a key requires moving the whole hand, do you have a hand switch afterwards to give your hand time to move back?

neighboring unbalance how often do you have to hit a neighboring key after a key forces your hand to move. Those require painful stretching of the fingers.

patterns how often do you have to do risky finger-transitions, minimizing transitions between ring finger - middle finger (dangerous for the sinews) and pinky - ring finger (typically slow).

manual penalty a table of bigrams given by positions of the keyboard that should be avoided

disbalance of fingers how well does the load on the fingers match, from pinky to index-finger 1.0, 1.6, 2.0, 2.0?

hand disbalance how even is the load distributed between the hands?

Aside from these movement costs there are three criteria for the ease of learning:

shortcut keys are the shortcut keys XCVZ easily reachable on the left side?

asymmetric bigrams when switching hands, how often do you have to hit a key that is not in a mirror-position to the first (an example mirror-position would be right pinky base-row to left pinky base row).

asymmetric similar keys are similar keys in similar position? For example p is a hard version of b and t is a hard version of d, so if p is below b, t should also be below d. For German: The umlauts äöü should be in the same relative positions to their base letters aou.

A final criterion is irregularity:

irregularity how regular is the ease of typing? If there are short segments of text that are really hard to type in between easy to type text, those cause the typing flow to stutter. The calculation uses the correlation between the two bigrams in each trigram as proxy: Multiply the cost of the two bigrams in a trigram, add those values together for all trigrams, and take the square root of the result.

An important and still imperfect part of this calculation is the corpus. We used a pre-aggregated corpus by the University of Leipzig that mainly built on newspapers as starting point and added with lower weight sources from email, wikipedia, a keylogger, chatlogs, the bash history, source-code, and Gutenberg texts. The exact definition can be found in the file ngrams.config.

In addition to direct bigrams, the optimizer splits trigrams and adds with lower weight the first and third letter as indirect bigrams. The same is done for bigrams with uppercase-letters (they are split into shift-first, first-second, shift-second) and special characters.

That's not all, but it covers the most important parts. You can find the details in the files config.py (weighting and basic definitions used) and layout_cost.py (the implementation).

Assigning weights with typing tests

These parameters were weighted against each other by creating three layouts for each parameter: One optimized with much higher weight for that parameter, one normal, one anti-optimized for this parameter by giving it negative weight. When increasing the weight of one parameter a lot, all other parameters become somewhat worse.

Then we did actual blind typing tests and ordered the layouts by quality. If the highly optimized layout performed worse, we checked which other parameter caused that. We then knew that the difference in this other parameter had a higher impact on typing quality than the parameter we actually optimized for.

That allowed us to nudge the different parameters into roughly fitting weight.

Finally we created many layouts and did practical typing tests with them, identified their weaknesses, matched them with optimization criteria and adjusted the weights for criteria to minimize the main problems we found. You can find some of the experiments and their conclusions in the notes in the folder Empirie.

With the explanations done, we can get to the results.

Layout Comparison Table

Update 2022: You can get more and updated statistics from `http://keyboard-layout-optimizer.herokuapp.com/`, a website with results from a rust-based reimplementation of the optimizer used here.

The following table compares the cost function for several layouts, both the aggregated cost (tppl: total penalty per letter) and the individual components. The layoutstring defines the layout using first the base row, then the top row, and finally the bottom row. Sorting by the layoutstring groups similar layouts together, because the base row has the biggest impact on typing.

Keep in mind that values for layouts generated with my optimizer (*mine*, *bone*, *cry*) and other layouts are not completely comparable, because other optimizers have different criteria for optimization. The uncertainty of the parameters to optimize could be higher than the difference between the layouts. Also keep in mind, that *bone* and *cry* were generated from older versions of my optimizer, so the criteria did not actually match those of *mine*, which might compensate for this effect and make the numbers comparable. *Mine* optimized for more criteria than the others. If those criteria are actually important (I think so: they are mostly those for ease of learning and regularity), this might make *mine* better. Note the high usage of "could" and "might" in this paragraph. We do not have strong estimates of the uncertainties of the parameters to optimize for.

Note: For ease of conversion from csv to table, the numbers here are not rounded but truncated: 12.8 is given as 12.

Table 1: Definition of different layouts with their total cost. Lower is better.

Layout-Name	layoutstring	**tppl**
mine	"crieo_mntsh-jluaq_wbdgyzß-vxüäö_pf,.k"	**11**
KOY	"haeiu_dtrnsf-k.o,y_vgclßz´-xqäüö_bpwmj"	**12**
AdNW	"hieao_dtrnsß-kuü.ä_vgcljf´-xyö,q_bpwmz"	**13**
bone	"ctieo_bnrsg-jduax_phlmwqß-fvüäö_yz,.k"	**13**
cry	"criey_ptsnh-bmuaz_kdflvjß-xäüoö_wg,.q"	**13**
Workman	"ashtg_yneoiä-qdrwq_jfupüöß-zxhcv_kl,.'"	**14**
Dvorak	"aoeui_dhtns–',.py_fgcrl/=-;qjkx_bmwvz"	**14**
Colemak	"arstd_hneio‘-qwfpg_jluy;[]-zxcvb_km,./"	**15**
Capewell	"aresf_ktnioö-.mydg_;wh,'äü-xczvj_bpluq"	**17**
Carpalx QGMLWY	"dstnr_iaeohü-qgmlw_byv;äöß-zxcfj_kp,.'"	**17**
Neo 2	"uiaeo_snrtd-xvlcw_khgfqyß-üöäpz_bm,.j"	**19**
QWERTZ	"asdfg_hjklöä-qwert_zuiopü+–yxcvb_nm,.-"	**37**

Table 2: Comparison of the properties of different layouts. Lower is better.

Layout-Name	**tppl**	key position	finger repeats	top <-> bottom	$(\text{rows}^2/\text{dist})^2$
mine	**11**	58	54	24	**105**
KOY	**12**	60	53	25	111
AdNW	**13**	**54**	55	24	113
bone	**13**	65	56	**22**	124
cry	**13**	67	55	23	114
Workman	**14**	97	**52**	25	121
Dvorak	**14**	78	55	28	119
Colemak	**15**	106	56	22	153
Capewell	**17**	76	60	20	148
Carpalx QGMLWY	**17**	154	54	26	143
Neo 2	**19**	199	53	24	180
QWERTZ	**37**	297	68	22	258

Table 3: Comparison of the properties of different layouts. Lower is better.

Layout-Name	**tppl**	handswitching	same hand after unbalancing	neighboring unbalance
mine	**11**	15	14	**10**
KOY	**12**	**7**	18	16
AdNW	**13**	**7**	16	16
bone	**13**	11	26	12
cry	**13**	15	14	11
Workman	**14**	28	**10**	**10**
Dvorak	**14**	9	24	21
Colemak	**15**	23	16	18
Capewell	**17**	22	19	24
Carpalx QGMLWY	**17**	15	23	29
Neo 2	**19**	11	22	14
QWERTZ	**37**	27	31	46

Table 4: Comparison of the properties of different layouts. Lower is better.

Layout-Name	**tppl**	patterns	manual penalty	disbalance of fingers	hand disbalance
mine	**11**	29	18	**7**	0.84
KOY	**12**	28	18	25	1.10
AdNW	**13**	37	18	14	1.10
bone	**13**	29	15	9	1.11
cry	**13**	25	**14**	12	**0.81**
Workman	**14**	30	15	18	1.46
Dvorak	**14**	26	23	45	1.92
Colemak	**15**	32	15	17	1.15
Capewell	**17**	57	16	46	0.51
Carpalx QGMLWY	**17**	24	19	18	0.83
Neo 2	**19**	**19**	14	20	0.26
QWERTZ	**37**	33	16	376	2.04

Table 5: Comparison of the properties of different layouts. Lower is better.

Layout-Name	**tppl**	shortcut keys	asymmetric bigrams	asymmetric similar keys	irregularity
mine	**11**	**0.0**	0.85	**11**	52
KOY	**12**	1.09	0.86	41	54
AdNW	**13**	1.09	0.85	53	63
bone	**13**	0.73	0.86	34	72
cry	**13**	0.73	0.85	63	61
Workman	**14**	0.73	0.88	69	**45**
Dvorak	**14**	1.46	0.86	48	57
Colemak	**15**	0.36	0.88	54	60
Capewell	**17**	0.36	0.86	73	59
Carpalx QGMLWY	**17**	0.36	0.87	72	64
Neo 2	**19**	0.36	**0.84**	41	97
QWERTZ	**37**	0.73	0.87	75	103

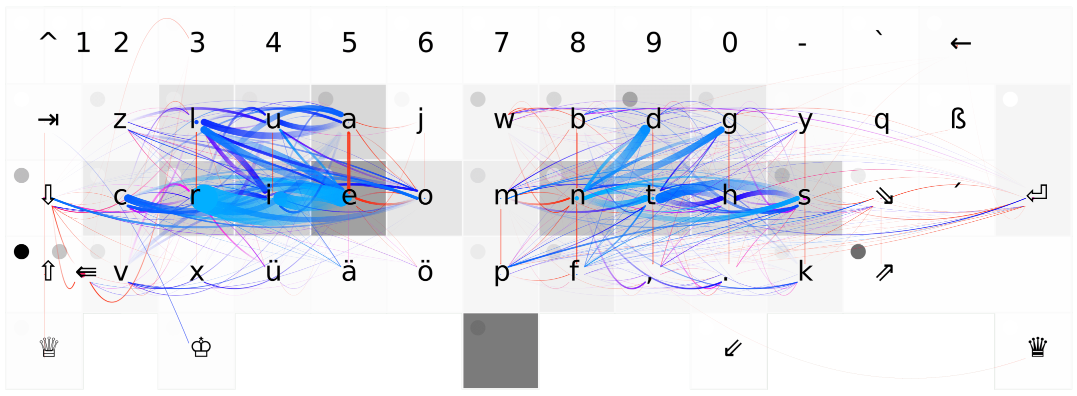
Layout Cost Graphics

The following graphics visualize the typing-properties of the layouts.

The opacity of the keys represents their frequency in the corpus. Darker keys are more common. The circle in the upper left of a key shows how often that key appears at the beginning of a word.

Each line between keys represents a bigram: Switching from one letter to the next. Its thickness shows how often that bigram occurs in the corpus we use (30% English, 60% German, 10% others). Its color shows the cost calculated for that bigram by the optimizer.

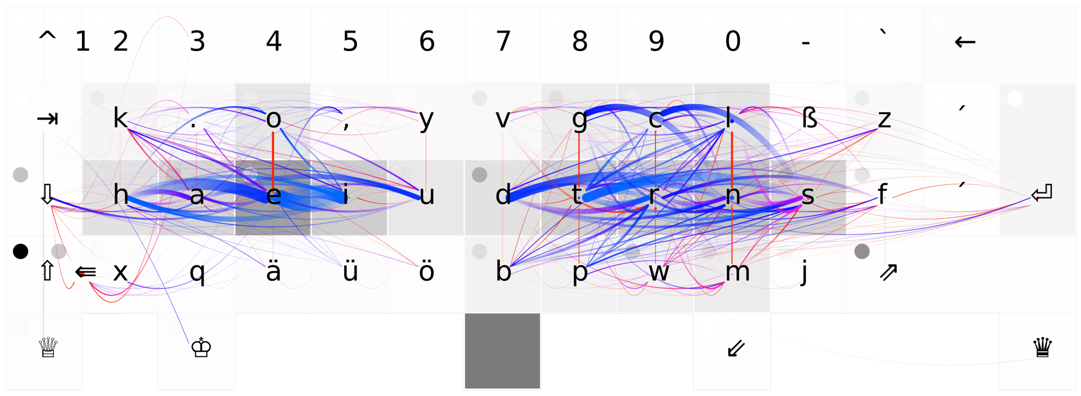
mine



zluaj wbdgyqß
crieo mnths⇘
vxüäö pf,.k

cost (tppl): 1102.0378762190887

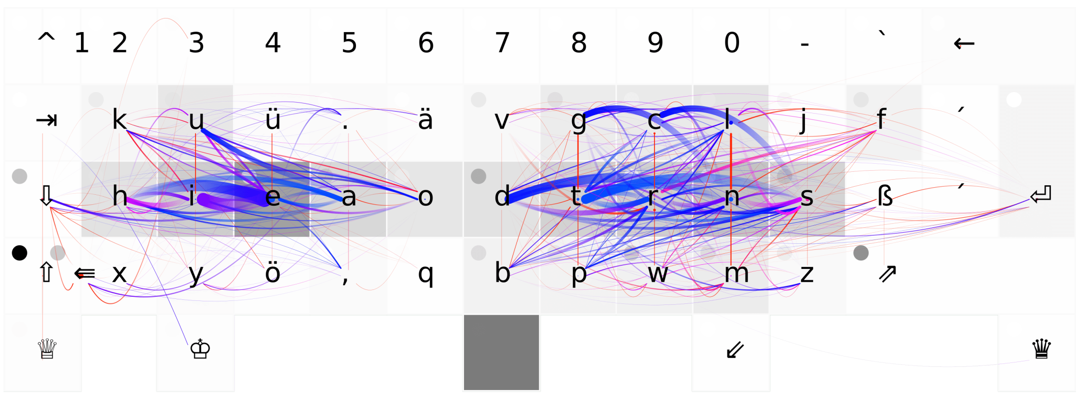
KOY



k.o,y vgclßz´
haeiu dtrnsf
xqäüö bpwmj

cost (tppl): 1240.540494787627

AdNW

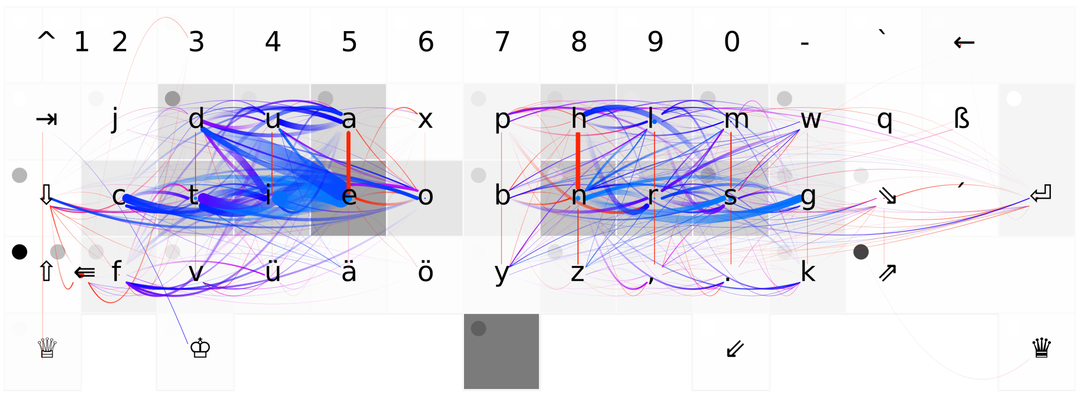


^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ k u ü . ä v g c l j f ´
⇩ h i e a o d t r n s ß ´ ⏎
⇧ ⇚ x y ö , q b p w m z ⇗
♕ ♔ ⇙ ♛

kuü.ä vgcljf´
hieao dtrnsß
xyö,q bpwmz

cost (tppl): 1308.3308542415928

bone

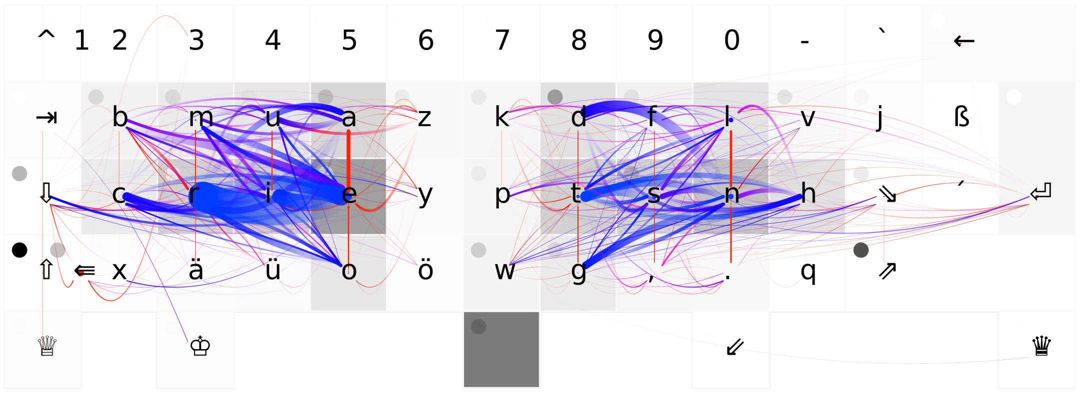


^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ j d u a x p h l m w q ß
⇩ c t i e o b n r s g ⇘ ´ ⏎
⇧ ⇚ f v ü ä ö y z , . k ⇗
♕ ♔ ⇙ ♛

jduax phlmwqß
ctieo bnrsg⇘
fvüäö yz,.k

cost (tppl): 1317.9949185140256

cry

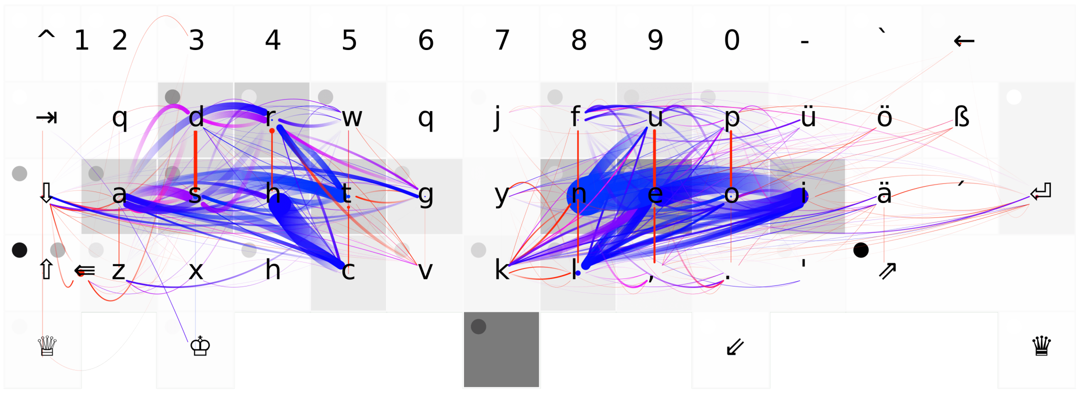


^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ b m u a z k d f l v j ß
⇩ c r i e y p t s n h ⇘ ´ ⏎
⇧ ⇚ x ä ü o ö w g , . q ⇗
♕ ♔ ⇙ ♛

bmuaz kdflvjß
criey ptsnh⇘
xäüoö wg,.q

cost (tppl): 1318.0320168766161

Workman

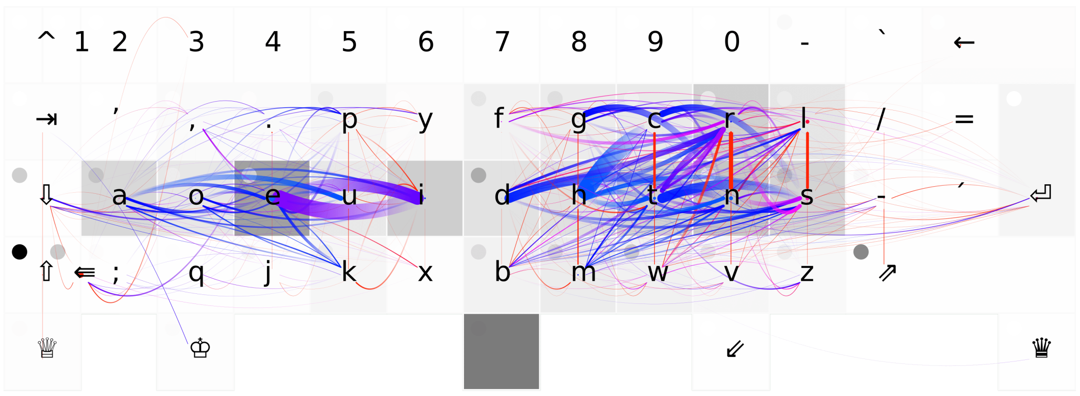


^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ q d r w q j f u p ü ö ß
⇩ a s h t g y n e o i ä ´ ⏎
⇧ ⇚ z x h c v k l , . ' ⇗
♕ ♔ ⇙ ♛

qdrwq jfupüöß
ashtg yneoiä
zxhcv kl,.'

cost (tppl): 1447.2371647588047

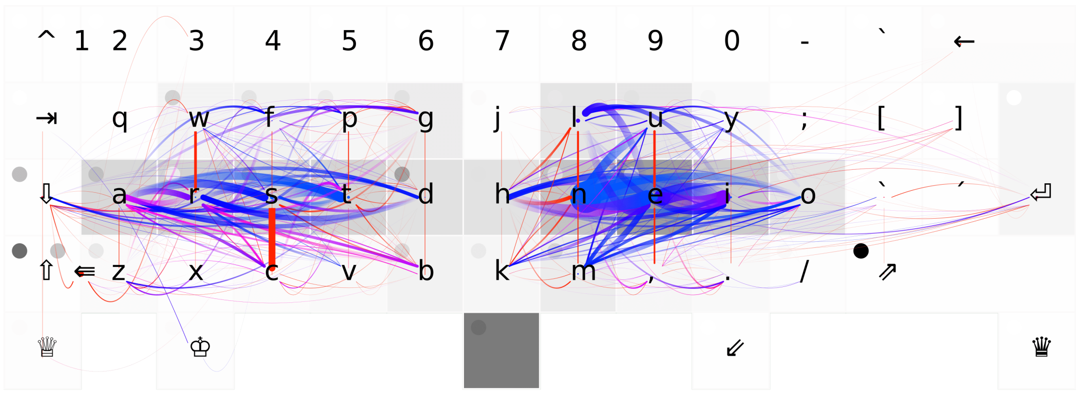
Dvorak


^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ ’ , . p y f g c r l / =
⇩ a o e u i d h t n s - ´ ⏎
⇧ ⇚ ; q j k x b m w v z ⇗
♕ ♔ ⇙ ♛

’,.py fgcrl/=
aoeui dhtns-
;qjkx bmwvz

cost (tppl): 1482.7640319024792

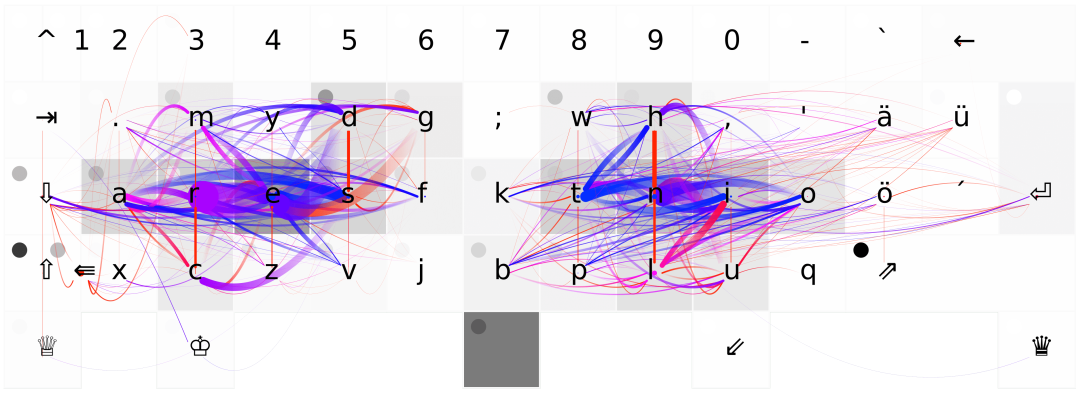
Colemak


^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ q w f p g j l u y ; [ ]
⇩ a r s t d h n e i o ` ´ ⏎
⇧ ⇚ z x c v b k m , . / ⇗
♕ ♔ ⇙ ♛

qwfpg jluy;[]
arstd hneio`
zxcvb km,./

cost (tppl): 1590.6264735545114

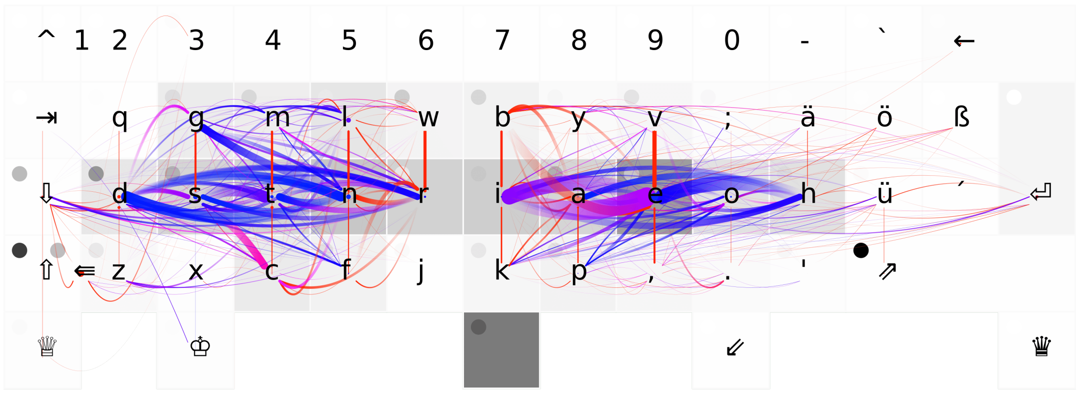
Capewell


^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ . m y d g ; w h , ' ä ü
⇩ a r e s f k t n i o ö ´ ⏎
⇧ ⇚ x c z v j b p l u q ⇗
♕ ♔ ⇙ ♛

.mydg ;wh,'äü
aresf ktnioö
xczvj bpluq

cost (tppl): 1714.890166136611

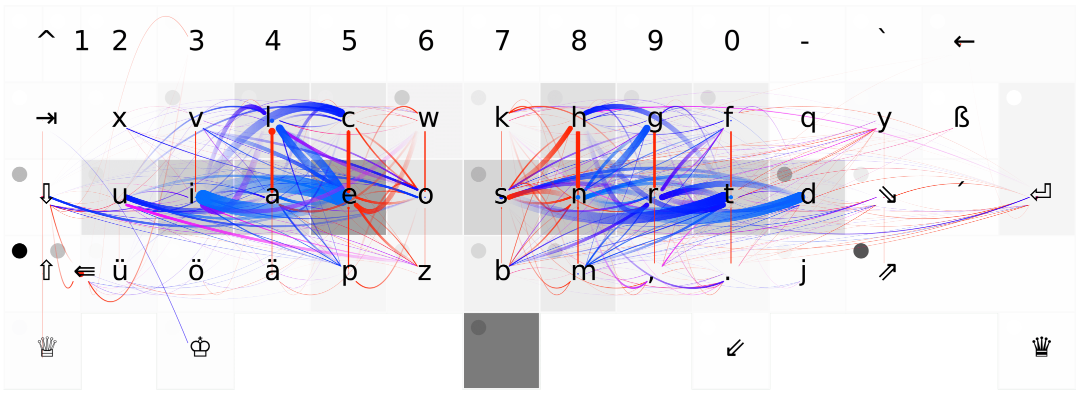
Carpalx QGMLWY


^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ q g m l w b y v ; ä ö ß
⇩ d s t n r i a e o h ü ´ ⏎
⇧ ⇚ z x c f j k p , . ' ⇗
♕ ♔ ⇙ ♛

qgmlw byv;äöß
dstnr iaeohü
zxcfj kp,.'

cost (tppl): 1772.0220676752203

Neo 2

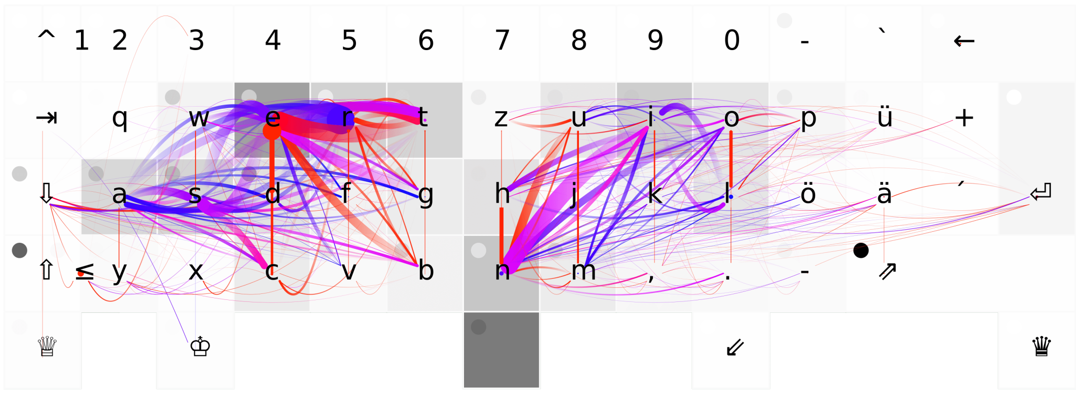


^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ x v l c w k h g f q y ß
⇩ u i a e o s n r t d ⇘ ´ ⏎
⇧ ⇚ ü ö ä p z b m , . j ⇗
♕ ♔ ⇙ ♛

xvlcw khgfqyß
uiaeo snrtd⇘
üöäpz bm,.j

cost (tppl): 1914.6490852416348

QWERTZ



^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ q w e r t z u i o p ü +
⇩ a s d f g h j k l ö ä ´ ⏎
⇧ ≤ y x c v b n m , . - ⇗
♕ ♔ ⇙ ♛

qwert zuiopü+
asdfg hjklöä
,.-

cost (tppl): 3705.598711312444

One-Hand layouts

Adjusting the optimization criteria allows optimizing layouts for one-hand usage.

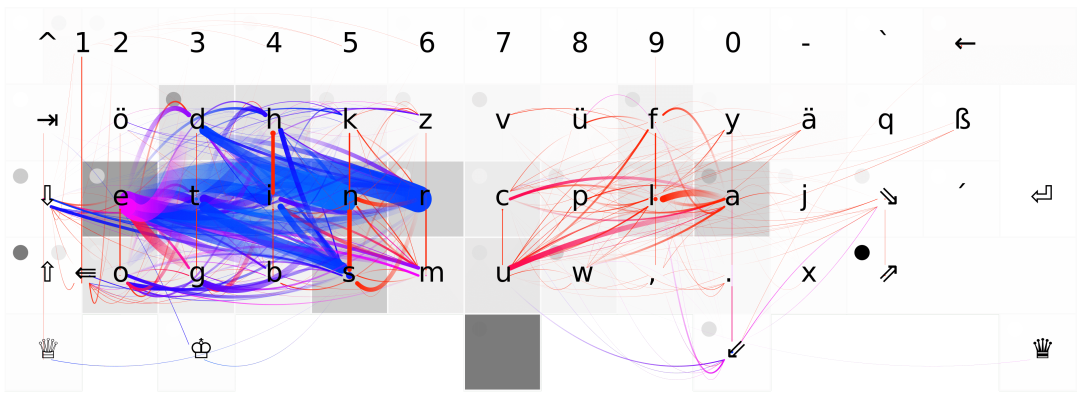
The following examples both focus on one hand, but also on longest possible sequences on one side. If you have to move your hand to the other side of the keyboard, the hand should ideally stay there for one or two more letters to reduce the disruption of the typing flow.

The left-hand-layout could actually come in handy if you often need to type with one hand at the mouse or if one of your hands is injured. Or with a kid on one arm (that's how I started experimenting with one-hand-layouts).

Learning a new layout typically takes a few weeks to three month to be back at your previous speed.

Both layouts are untested, though (if you test one of them, please tell me your results!).

Left-Hand Layout

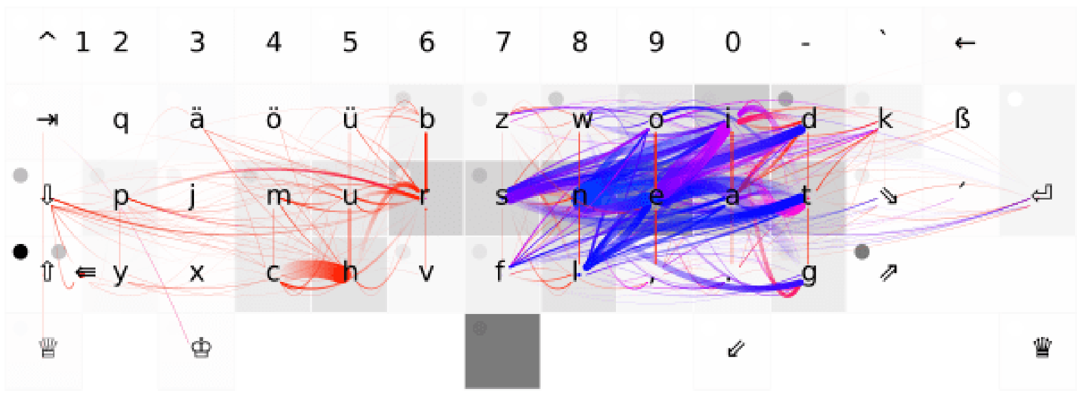

^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ ö d h k z v ü f y ä q ß
⇩ e t i n r c p l a j ⇘ ´ ⏎
⇧ ⇚ o g b s m u w , . x ⇗
♕ ♔ ⇙ ♛

ödhkz vüfyäqß
etinr cplaj⇘
ogbsm uw,.x

cost (tppl): 1661.5063583944648

You can test the left-hand-layout on GNU Linux using its xmodmap. I switch to lv layout before xmodmap: `setxkbmap lv && xmodmap tir-links.xmodmap`

Right-Hand Layout


^ 1 2 3 4 5 6 7 8 9 0 - ` ←
⇥ q ä ö ü b z w o i d k ß
⇩ p j m u r s n e a t ⇘ ´ ⏎
⇧ ⇚ y x c h v f l , . g ⇗
♕ ♔ ⇙ ♛

qäöüb zwoidkß
pjmur sneat⇘
yxchv fl,.g

cost (tppl): 1259.213456765156

You can test the right-hand-layout on GNU Linux using its xmodmap. I switch to lv layout before xmodmap: `setxkbmap lv && xmodmap neat-rechts.xmodmap`

Conclusion

Why all that? After typing for 7 years with the cry-layout, I don't ever want to go back to QWERTZ. It feels awkward and clumsy, always having to do unnecessary extra steps.

If you want to give this a try, have a look at neo-layout.org, use a translator if necessary. Join the German Neo-Community. Get information about other layouts.

If you're from Bulgaria, you already use an ergonomic keyboard. We in Germany envy you for that.

If you write other languages besides English and German, or you never write German, you could use the optimizer with a different corpus. Have a look at evolution.py and the README.md. If you can only use one hand, you might benefit from giving the onehandright branch a try, and maybe adapting it for the left hand (if that's your good hand). One example of a one-hand-layout is neat-rechts (xmodmap).

I hope that `mine` will be the keyboard layout I stick to for the rest of my life — or at least for as long as I can use both hands to type. And I hope others can pick up where I started and generate more optimized layouts. And maybe this can become a base for a better standard layout for the EU so we no longer have to look at Bulgaria with envy but can instead celebrate them for spearheading an evolution of typing in Europe.