Using Cell Phone Keyboards is $\mathcal{NP}$ Hard

Peter Boothe

peter.boothe@manhattan.edu
Mathematics and Computer Science
Manhattan College

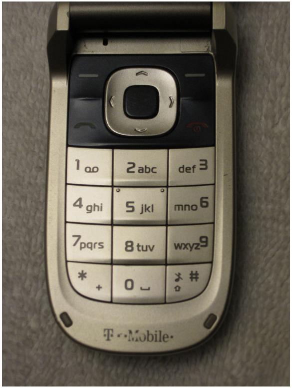
2 June 2010
FUN With Algorithms

So many letters . . .

- People want to use phones to send text messages
- English has 26 letters, and each letter has a lower-case and capital form, and we should not forget numbers, puctuation, and whitespace characters.
- Other languages have even more![1]

[1] Can you name a language with fewer letters than English?

. . . so few keys



- A computer keyboard usually has more than 100 keys.
- A phone keyboard has, depending on the model, as few as 8 keys which may be used to input letters.

Two Input Methods

The old default mapping of numbers to letter sequences:

$$\begin{array}{lll} 1 \rightarrow [] & 2 \rightarrow [a,b,c] & 3 \rightarrow [d,e,f] \\ 4 \rightarrow [g,h,i] & 5 \rightarrow [j,k,l] & 6 \rightarrow [m,n,o] \\ 7 \rightarrow [p,q,r,s] & 8 \rightarrow [t,u,v] & 9 \rightarrow [w,x,y,z] \end{array}$$

Standard Press the appropriate key until your letter appears. e.g. Press 8 two times to get a ‘u’

T-9 Load a word-frequency dictionary for the language into the phone, and only require that people type one key for each character. Display the most popular word with a given keypress sequence.

Our question

Both of these input methods assume the old default mapping. If we throw that assumption out, then, for each typing method:

What is the mapping of numbers to letters that minimizes the number of keypresses required to type a message?

Our question, formalized

INSTANCE A number of keys, k, an alphabet A, an input method $\mathrm{IN} : (\mathit{word}, \mathit{partition}) \rightarrow \mathbb{N}$, and a set of words and associated frequencies W.

QUESTION What is the partition P of A into k sequences which minimizes

$$\sum_{(w,f)\in W} f \times \mathrm{IN}(w, P)$$

?

The Standard Input Method

- Press a key as many times as needed to made the desired letter appear e.g. "fun" is 3338866, requiring a total of 7 button presses
- Word-independent
- Easy to understand
- Inefficient e.g. $7 \longrightarrow [p, q, r, s]$ means you have to go through 'q' to get to 'r' or 's'

Greedy Algorithm

- Sort the alphabet by letter frequency.
- Assign letters to keys in round-robin order.
- Proven optimal via an exchange argument (see paper)
- Rearranges the letters on the keys
- Possibly a very confusing layout

Legacy-Preserving Greedy Algorithm

- Sort each key's letters by frequency. Reorder the letters on each key in decreasing frequency order.
- Using the British National Corpus, this reduces button presses by 28.43%
- Less confusing than remapping the whole keyboard
- Just fixing the 7 key will save you $\sim$10%

An Open Letter

Dear Cell Phone Manufacturers,

Please change your keyboard layout to:

$$
\begin{array}{lll}
1 \rightarrow [] & 2 \rightarrow [a,c,b] & 3 \rightarrow [e,d,f] \\
4 \rightarrow [i,g,h] & 5 \rightarrow [l,k,j] & 6 \rightarrow [o,n,m] \\
7 \rightarrow [s,r,p,q] & 8 \rightarrow [t,u,v] & 9 \rightarrow [w,y,x,z]
\end{array}
$$

This new layout will require your customers to type 28% fewer keystrokes when sending an SMS message, but will not break such famous numbers as 1-800-FLOWERS, 1-800-MARINES, or PA-6-5000. Or at least fix the 7 key.

Sincerely,
Peter Boothe

The T-9 Input Method

- Per-word, not per-character
- One key per letter, display the most popular word with that key sequence e.g. 'home' is 4663.
- In case of collision, cycle through all words with that key sequence in order of popularity using * e.g. 'home', 'good', 'gone', 'hood', 'hoof', 'hone', and 'goof' are all 4663, so 'goof' is 4663******
- Two words with the same sequence are **t9onyms**

Can we minimize the expected number of * key presses?
or, equivalently,
Can we minimize the expected number of t9onyms?

Minimizing T-9 is $\mathcal{NP}$-complete

- We have sequences in the problem (the words in the dictionary)
- We have partitions in the problem (the assignment of letters to keys)
- Couldn't find a NPC problem with both, so invented one
- Proof is via reduction from UniquePathColoring

UniquePathColoring

Instance A graph $G = (V, E)$, a set of paths P, and a parameter k.

Question Is there a k-coloring of V such that every path in P has a unique coloring?

Note that this problem meets our criteria of having sequences (the paths of P which correspond eventually to the words in W) and partitions (the colors of the graph, which correspond eventually to the key assigned to each letter). UniquePathColoring is equivalent to asking if there is an assignment with no t9onyms.

UniquePathColoring is $\mathcal{NP}$-Complete

Reduction from GraphColoring.

Take the $G = (V, E)$ and k from GraphColoring, and construct G'.

$$
\begin{aligned}
G' = \; (\quad & V \cup \{0, 1\}, \\
& E \cup \{(1,1), (0,1), (0,0)\} \cup \{(v, 0)\ \forall v \in V\} \\
) \quad &
\end{aligned}
$$

UniquePathColoring is $\mathcal{NP}$-Complete (2)

Next, uniquely number the edges in E. Construct P by making paths p_i for each edge (u, v) in E.

$$
\begin{aligned}
p_i = \quad \{ \quad & [u, 0, \mathrm{b}_1(i), \mathrm{b}_2(i), \ldots \mathrm{b}_{\lceil \log_2 i \rceil}(i)], \\
& [v, 0, \mathrm{b}_1(i), \mathrm{b}_2(i), \ldots \mathrm{b}_{\lceil \log_2 i \rceil}(i)], \\
\} &
\end{aligned}
$$

where $\mathrm{b}_j(i)$ is the jth binary digit of i.

$$
P = \bigcup_{1 \le i \le |E|} p_i
$$

UniquePathColoring is $\mathcal{NP}$-Complete (3)

Does there exist a UniquePathColoring of G' with paths P using no more than k colors? If so, then the colors assigned to the vertices in V form a k-coloring of G.

Does there exist a k-coloring of G? If so, we can construct a UniquePathColoring of G' and P using k colors — simply choose two distinct colors for the new vertices 0 and 1.

$\therefore$ UniquePathColoring is $\mathcal{NP}$-complete

Minimum-T9onyms is $\mathcal{NP}$-complete

The decision problem Minimum-T9onyms:

Instance Alphabet A, set of words and associated frequencies W, number of keys k, number t.

Question Is there a partition of A into k sets such that

$$t \geq \sum_{(w,f)\in W} f \times (\mathrm{len}(w) + \mathrm{order}(w, P, W, f)) \ ?$$

Note that this exactly conforms to our problem specification from the beginning.

Minimum-T9onyms is $\mathcal{NP}$-complete (2)

If we require that there be no t9onyms, then this simplifies to

$$t \geq \sum_{(w,f)\in W} f \times \mathrm{len}(w)$$

or simply a requirement that order() be zero for all words in W.

Minimum-T9onyms is $\mathcal{NP}$-complete (3)

The reduction from UniquePathColoring...

UniquePathColoring	Minimum-T9onyms
Instance Graph $G = (V, E)$, paths P, and k colors	Instance Alphabet V, words P all with frequency 1, number of keys k, and $t = \sum_{p \in P} \mathrm{len}(p)$
Question Does there exist a k-coloring of V such that every path $p \in P$ has a unique coloring?	Question Does there exist a k-partition of V such that every word $p \in P$ has order() zero?

$\square$

Baseline

Our baseline comparison was the number of keystrokes required to enter the entire British National Corpus using T-9 with the standard layout.

T-9, with the default layout, requires 443,374,079 keypresses to enter the entire BNC.

Enumerating All Alphabetic Partitions

If we restrict ourselves to only the partitions which keep the letters in alphabetic order, then we can enumerate all $\binom{25}{7}$ partitions and test them over the course of a few days on a modern computer.

We find that the best partition is

$$\{\{ab\},\{cde\},\{fghij\},\{klm\},\{nop\},\{qrs\},\{t\},\{uvwxyz\}\}$$

This partition requires 442,717,436 keypresses, a savings of 656,643 keypresses, or 0.14%.

Genetic Algorithm Results

A very simple genetic/stochastic algorithm was run for a few days. The best solution found was

$$\{\{be\}, \{cdl\}, \{hs\}, \{mxz\}, \{afgy\}, \{jot\}, \{npuv\}, \{ikqrw\}\}$$

This required 441,612,049 keypresses, a savings of 1,762,030 keypresses, or 0.40%.

Future Work

- Predictive T-9
- Other input methods
- When T-9 gets a misspelled word, it is very bad and confusing. Can we minimize not just t9onyms, but also the number of words that have a cell-phone Levenshtein distance of 1? of 2? of k?
- We completely ignored the pauses that the standard method requires when typing two letter in a row which involve the same key
- Completely solving the particular case of T-9 with the English language and an 8-key keyboard
- Universal keyboards — fixed labelings which are good for multiple languages using the standard typing method (thanks Markus Holzer!)

Take-away Messages

- Cell phone keyboards can provide algorithmic fun
- The standard input method can be optimized with the greedy algorithm
- The T-9 input method is NP-Complete to optimize
- The existing keyboard seems alright for T-9, but can be improved by 28% for the standard input method without breaking legacy advertising
- Cell phone keyboards are not just hard to use, they can also be $\mathcal{NP}$-hard to use

Thanks!

Questions?