Towards an Empirical Cost Model for Mental Password Algorithms

Enka Blanchard
Digitrust, Loria, Université de Lorraine
Vandoeuvre-lès-Nancy, France
Enka.Blanchard@gmail.com

Ted Selker
Salker Design Research
Palo alto, CA, USA

Florentin Waligorski
Observatoire de Paris
Paris, France

ACM ISBN 978-1-4503-6819-3/20/04.
http://dx.doi.org/10.1145/3334480.3382990

Abstract

Reliance on technology has diminished our use of mental computation. However, mental computation's inherent privacy features are becoming central to new research on creating more secure and usable passwords than one gets with approaches such as password managers. This work empirically studies the validity of cognitive assumptions relative to mental computation for making codes like passwords, using as a starting point password algorithms and a cost model for mental computation developed by Blum and Vempala. Through a study on 126 participants, we refute some of their model's assumptions, and introduce evidence of behaviours where human computing costs behave counter-intuitively. We also identify three empirical questions around symmetry, repeatability, and distribution of costs whose resolution would allow the development of more predictive cognitive computation models. This would then allow the efficient creation of better security algorithms.

Author Keywords

Psychometrics; Password; Mental Computation; User Study

CCS Concepts

•**Human-centered computing** $\rightarrow$ **User models;** Empirical studies in HCI; •**Applied computing** $\rightarrow$ *Psychology;*

Operation	Cost
Access item # 1	1
Access item # 2	2
Retrieve pointer	1
Follow pointer	1
Reset pointer	1
Apply a map	1
=, + and x (mod 2,3,4,5,9,10)	#digits created
=, + and x (mod 11)	1+#digits created

Table 1: A summary of Blum and Vempala's computational cost model for passwords includes short-term memory as a 2-item stack, and long-term memory as a set of pointers. Resetting the pointer means going to either the start or the end of the relevant memory item. Operation costs only apply to single or double digit numbers. Applying a map generally corresponds to a letter-to-number or letter-to-letter map.

Introduction

In a series of papers Blocki, Blum, Datta and Vempala introduced a series of mental algorithms meant to replace password managers, and also a general model for mental computing [4, 5, 6, 7, 23]. This model featured a stack, a set of pointers, and costs for basic operations, allowing the formalisation of already existing informal algorithms [20]. More algorithms have since extended this research [25], including a method that produced 42-bit secure independent passwords in under a minute while avoiding mental arithmetic [3].

The development of new mental algorithms is hampered by the high cost linked to testing. Each new algorithm (and every variant) requires an independent usability study to ascertain its performances when compared to other solutions. This would be made easier by having a cost model for mental algorithms that would give the distribution of how time-intensive it is for individuals. We identify three obstacles must be overcome to create an accurate cost function *C*.
Repeatability. We can't assume that $C(A;A) = 2\times C(A)$ for an algorithm A, due to both habituation and mental fatigue.
Symmetry. We can't assume that the $C(A;B) = C(B;A) = C(A){+}C(B)$ for the concatenations of two sub-algorithms A and B, as task switching might create non-symmetric costs.
Average cost versus distribution. *C* can be highly dependent on the human considered. This is why, assuming there are clusters of humans with similar performance on similar problems, a distribution of time (instead of an expected time) might be obtainable and more informative.

Cognitive science approaches. Some aspects of mental algorithm performance has been studied extensively with approaches from both psychometry and neuroscience, with studies on learning and memory abilities for a variety of stimuli (such as words, sentences, pictures) [24, 17], mnemonics [2], and comparisons with other primates [18].

A second focus has been the arithmetical capabilities of the average human, with correlations between arithmetic, algorithmic, and linguistic skill [14, 15]. It also led to the exploration of neurological bases for our representation of numbers [1, 9, 12].

Differences between demographic groups have also inspired extensive studies, especially for arithmetic abilities of people with maths anxiety [13], with short-term memory problems [11], and across cultures [22] and age groups [10].

Blum-Vempala model. Surprisingly little has been done in the search for a general cost model for human computing that would accurately predict, quantitatively, the difficulty and time cost to compute mental algorithms. So far, the main model used in security was introduced by Blum and Vempala in 2015 [7], and refined in [6]. This model, shown in Table 1 was meant to give a rough expectation of time taken but wasn't experimentally validated. The authors also insisted on keeping a low total cost for mental password algorithms; proposing a maximum cost of 12 as a recommendation to cap time under 40 seconds, assuming that an average individual spends 2 to 3 seconds on a task of cost 1.

The model's costs might be reasonable, but are its assumptions correct? For example, it has no place for queries of variable complexity (e.g. getting the n-th letter in the alphabet, which seems easier for $n = 1$ than for $n = 13$, but which they assume has a uniform cost). We show here the early results of a study on 126 participants to expose mechanisms involved in creating a cost function for mental algorithms. Among other results, we show that accessing a mental map is far from uniform, with a difference of one order of magnitude depending on the element accessed. Arithmetic operations also behave differently, with $n + m$ apparently costing not $\theta(\log(n) + \log(m))$ but closer to $\theta(n + m)$.

This work is exploratory in nature. While our work confirms — and refutes — some of the assertions made by the published model, its goal is to show the need for and value of a rigorous and empirically tested model, to avoid statistical mistakes that can stymie research replication [8].

Design of the experiment

The goal of this study was to give preliminary answers to some of the following questions and offer insight to researchers studying and proposing new cognitive algorithms. The main goals were the following:

- Get baseline time cost distributions for elementary operations considered in a way that allows comparison between groups and user groups.
- Check whether accessing the n-th element in an array is done in constant time or not, as asserted by Blocki et al [4], and check the effects of repeated accesses.
- Check whether the costs commute: is it easier to perform an addition after a multiplication or the other way around?
- Check the independence of users' performance in elementary linguistic and arithmetic tasks. This is especially relevant for password algorithms as having clusters with different abilities would motivate the need for a set of tailored algorithms instead of a general one.

Demographics

126 participants[1] were recruited through a mix of snowball sampling on social networks [16] and referral through a website that indexes psychological experiments [19], with data from four being discarded as they skipped most of the questions. As a high number of participants were expected to be French, two versions of the experiment were developed and participants could choose their preferred version on the first page. 85 were native English speakers, 28 were French, and the rest spoke 10 different languages. Ages went from 15 to 63 with a median at 23 and an average at 27.

[1]As the country where the experiment was organised does not have institutional review boards, we followed the guidelines for human subject studies enforced in the US as best practices. Participants did not belong to protected groups, were told they could leave the trial at any time, and no personal or identifying information was collected besides age and main language spoken.

Protocol

The participants interacted with the system through a webpage that recorded their inputs as well as the times to type a first character and to complete each query. For tasks where the answer is a number or a word, the measures and comparisons were all made on the basis of the first character typed — to compensate for answers of different lengths among other things. To attenuate the delay due to reading and understanding the questions, complex instructions were shown on an example before the question was asked.

The experiment had 9 sections of varying length:

1. Participants were initially told their rights and asked basic demographic questions.
2. They were told to learn a four-word sentence, type it, and type it again at the end of sections 3, 4, and 5. They were shown their sentence if they made a mistake.
3. They were given three single-digit additions, single-digit multiplications, and double-digit additions. Then two multiplication on lower double-digit numbers and four remainders of double-digit numbers divided by 3,5,9 and 11.
4. They had to produce a series of three words starting by t, b and a (three different questions), like tortoise, blueberries and Austria. In a between-groups approach, half were asked for an animal, fruit and country, while the other half were asked to type in any word

starting by the same letter.

5. They had to give the n-th letter in the alphabet, for four different letters in a row. They were then asked the reciprocal question, also four times. The numbers were written with Arabic numerals except for one group for which they were spelled out. They were then successively given three words and asked to write the letters in alphabetical order. Finally, they had to say what was the n-th month of the year (for two different months), followed by two reciprocal questions.
6. Participants had to find a given letter in either a sentence on screen — or the one they had memorised — and write the two letters that followed the given letter.
7. They had to type the words they had chosen earlier.
8. Participants had to apply an arithmetic algorithm (involving additions and multiplications), then a linguistic algorithm (e.g. finding letters in sentences), and finally a algorithm mixing both kinds of operations.
9. Finally, participants were asked whether they had used a pen and paper, and for which tasks.

Between-group design was used in multiple ways in this experiment. One was that sections 3, 4 and 5 were given in a random order. For most of the questions, participants were divided into 3 to 5 groups with different but structurally identical questions given to each group.

For sections 3 and 5, one group was given random numbers or letters, while the others were given predetermined questions. This was to ensure that enough data would be collected to establish statistically significant results on certain properties, while maintaining the possibility that, if enough people participated, there would be data on more than a few specific points. For example, in section 3, a quarter of the participants started with the question "2+2=?", another quarter with "9+8=?", and the rest with random pairs of digits.

Preliminary results

One difference with using these techniques in the wild is that in the experiment participants had to read the instructions and assimilate them before performing the trials, adding some noise to the data. We think this strengthens the results — as this added noise was not enough to make the effect disappear. As the experiment was online and not in a controlled environment, we couldn't prevent people from taking a break (and some participants did take to a 10 minute break). The p-values indicated correspond to one way ANOVAs. No significant effects were observed between the two main language groups.

Access time in a list

The first main question was the veracity of the constant access time in a list assumption in the published model, with two different tests for this in section 5. Our expectation was that the first few elements of the list might get accessed in constant time, with a sub-linear increase until close to the end of the list, where it would go back down as people chose to work from the end of the list instead.

This was partially shown to be true: on the first question, there was no significant difference between asking for the letter "A" or the letter "B", with both average and median times between 2.6s and 2.9s. However, "D" is quite slower than "A" (3.4s on average), with $p < 0.0003$. This difference got absorbed by the second question with no statistically significant differences between "K", "M" and "N" (respectively 8.0, 8.2 and 8.3 seconds on average).

The strongest result came from the third question. All the users who were asked to give the 13th letter ("M") had previously had to give the 14th ("N") in the third question, and did so in 2.2s on average. The ones who were previously asked to give the 14th letter were now asked for the 13th, and did so in 3.6s on average. Getting the next element in the list is then much easier than getting the previous one ($p < 0.0005$). The speed ratio is at least a factor 1.5, but might even be close to 3, assuming a 1s reaction delay (reasonable as less than 1% of answers were faster than 1s).

Still on the third question, the participants who had had the 11th letter ("K") and had to give the 18th ("R") afterwards did not benefit from their previous computations and took an average of 12.0s. This is confirmed by the fact that getting the 22nd and 23rd letters in the last question also took 14.0 and 14.6 seconds, with no visible priming effect from what participants had to answer just before. Although the sample size for those letters was small, we could still observe speed increase for the last 3 letters of the alphabet, compared to the 3 preceding characters in the alphabet, suggesting that some people proceeded backwards, with an average of 10.1s, compared to 14.9s on average for the three preceding letters ($p < 0.05$).

Similar results can be shown for character-to-number maps. For example, giving the number corresponding to "D" took an average of 4.8s, as opposed to 3.9s for "A" on the first question, and 7.2s for "K" on the second question ($p < 0.03$ and $p < 10^{-4}$ respectively).

We confirmed those intuitions on month/number queries. When initially asked for either the first or the fifth month of the year, participants were much slower in the latter case (2.8s and 4.6s on average, with medians of 2.4s and 3.8s, $p < 0.10^{-4}$). In the second question, when asking for the tenth month, there was no statistical difference between the groups who had been asked for the first or the fifth previously. The reciprocal question also featured speed differences, with averages of 2.1s to say that February was the 2nd month, versus 4.0s for July being the 7th ($p < 005$).

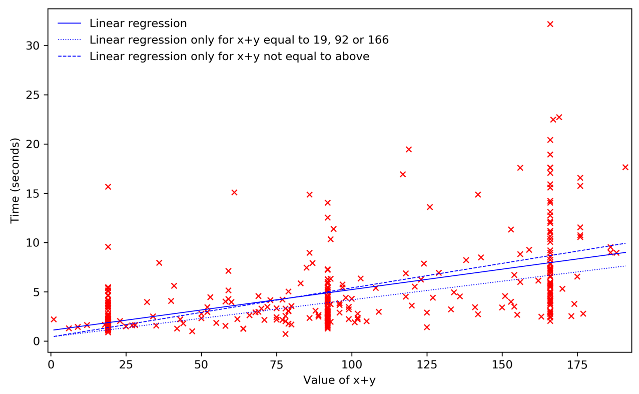

Figure 1: Time taken by participants to compute "x+y" as a function of the expected value . Two-thirds of the participants were given the tasks of computing "12+7", "79+87" and "87+5" in various orders, with the rest having random 2-digit additions (one of the linear regressions shown focuses on this group, whereas another ignores it, with little difference in the coefficients). There is a tendency to increase at least linearly, not the logarithmic growth in the published model.

Arithmetic operations

We observed a large speed increase between the first and second question — both being single-digit additions — with the average time going down from 2.7s to 1.8s ($p < 10^{-4}$), which we attribute to switching costs between tasks before the first question and the first question. Discarding this first question, we observed no statistical difference between the cost of doing "2+2" and "4+5" on the second question. However, we saw a difference between "4+5" and "8+9" (1.9s vs 2.3s, $p < 0.05$).

The same effect can be seen with multiplication: results with a single digit were computed faster than ones with double digits (averages of 1.6s versus 4.8, with $p < 2 \times 10^{-4}$).

Algorithm	Time
Start with 2. Multiply by 2. Add 4. Divide by 4.	8.3s / 6.6s
Start with 2. Add 2. Add 4. Divide by 4.	7.7s / 7.3s
Start with 2. Multiply by 2. Add 4. Subtract 6.	7.7s / 6.4s
Start with 20. Add 20. Add 40. Divide by 4.	11.1s / 7.0s
Start with 6. Add 6. Add 12. Divide by 4.	10.0 / 7.9s

Table 2: Average and median execution time of each arithmetic algorithm.

The previous results could be seen as a partial confirmation of the published model, but detailed analysis and extension of the addition to 2-digit numbers show that time cost is more probably linearly dependent on the result, with the spread increasing with the complexity, as shown in Figure 1.

To test how the operation complexity depended on the context and previous operations, participants had to enter the result of one of five quasi-equivalent arithmetic algorithms — going through the same set of numbers with different operations — shown in Table 2. The first and the third, with their different operations, had statistically equivalent performance to the second, but had a much wider spread. All three were also faster than algorithms 4 and 5 ($p < 0.02$), which are based on algorithm 2, except that all values are multiplied by 10 (for #4) or 3 (for #5).

Discussion and future work

Exploring the cognitive issues surrounding security practices is overdue. This paper presents empirical testing of assumptions researchers have made regarding cognitive issues that arise when mentally creating strings of characters for passwords. The results of of our experiments show that what character you retrieve, what computation you do and how many of them you do have real impacts on each other.

Some assumptions that led to the creation of the published model were not validated, most importantly the idea that humans can assign a map of numbers to letters and use it quickly as a hash function does not work. Instead, the computing time depends strongly on position; there is a large difference between going forward or backward in the list, and there is very limited re-use of already computed elements. From an ecological standpoint, the published model assumed that people would have a training period, although one limited to a few dozen minutes. This is not compatible with its predictions, as it would either have to compensate for decades of semi-regular use, or the "time to access" cost only concerns new maps/functions that people haven't practised before. In such a case, it would stand to reason that most people would get better performances on the functions on which they are trained since childhood, and one needs to separate the costs depending on the kind of function. In both cases, the model requires changes, especially since it was originally developed for critical algorithms where a factor 2 or 3 in the cost matters a lot in terms of wide-scale human adoption of the newly developed methods.

This late-breaking work shows some surprising elements. Besides the very limited re-use of precomputed elements, operating the algorithm in section 8 with multiples of 10 takes more time than with multiples of 3 (despite being with double digit numbers in both cases). There are also quite a few questions that will require further analyses: what is the time cost of memory recall (sections 4 and 7), of letter manipulation (sections 5 and 6), of letter searching in a text (sections 6 and 8), and what additional cost is linked to alternating between linguistic and arithmetic tasks (section 8).

The goal of this experiment was to underline the questions at the heart of modelling the cost of human computation, and to provide an empirical basis for a new, more comprehensive model. Such a model should not only include time costs (although they are the easiest to measure), but also perceived cost, linked to the complexity and repetitiveness of the tasks involved. Moreover, just as pre-WW2 cryptosystems relied on a mix of manual labour and precomputed codebooks [21], new algorithms based on the privacy-enhancing properties of tool-aided mental computing could form the basis of new security paradigms.

Funding: this work was supported partly by the french PIA project “Lorraine Université d'Excellence”, reference ANR-15-IDEX-04-LUE.

REFERENCES

[1] Mark H. Ashcraft. 1992. Cognitive arithmetic: A review of data and theory. *Cognition* 44, 1-2 (1992), 75–106.

[2] Francis S. Bellezza. 1987. *Mnemonic Devices and Memory Schemas*. Springer New York, New York, NY, 34–55. DOI: http://dx.doi.org/10.1007/978-1-4612-4676-3_2

[3] Enka Blanchard, Leila Gabasova, Ted Selker, and Eli Sennesh. 2019. Cue-Pin-Select, a Secure and Usable Offline Password Scheme. (2019). https://hal.archives-ouvertes.fr/hal-01781231

[4] Jeremiah Blocki, Manuel Blum, and Anupam Datta. 2013. Naturally rehearsing passwords. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 361–380.

[5] Jeremiah Blocki, Manuel Blum, Anupam Datta, and Santosh Vempala. 2017. Towards Human Computable Passwords. *8th Innovations in Theoretical Computer Science Conference – ITCS 2017* (2017).

[6] Manuel Blum and Santosh Vempala. 2017. The Complexity of Human Computation: A Concrete Model with an Application to Passwords. *CoRR* abs/1707.01204 (2017). http://arxiv.org/abs/1707.01204

[7] Manuel Blum and Santosh Srinivas Vempala. 2015. Publishable humanly usable secure password creation schemas.. In *3rd AAAI Conference on Human Computation and Crowdsourcing*.

[8] Denny Borsboom. 2006. The attack of the psychometricians. *Psychometrika* 71, 3 (2006), 425.

[9] Brian Butterworth. 1999. *The Mathematical Brain*. Macmillan.

[10] Brian Butterworth. 2002. Mathematics and the Brain. (2002). Opening Address to The Mathematical Association.

[11] Brian Butterworth, Lisa Cipolotti, and Elizabeth K. Warrington. 1996. Short term Memory Impairment and Arithmetical Ability. *The Quarterly Journal of Experimental Psychology Section A* 49, 1 (1996), 251–262. DOI: http://dx.doi.org/10.1080/713755603

[12] Brian Butterworth, Marco Zorzi, Luisa Girelli, and A. R. Jonckheere. 2001. Storage and retrieval of addition facts: The role of number comparison. *The Quarterly Journal of Experimental Psychology Section A* 54, 4 (2001), 1005–1029. DOI:http://dx.doi.org/10.1080/713756007 PMID: 11765730.

[13] Hyesang Chang, Lisa Sprute, Erin A. Maloney, Sian L. Beilock, and Marc G. Berman. 2017. Simple arithmetic: not so simple for highly math anxious individuals. *Social cognitive and affective neuroscience* 12, 12 (2017), 1940–1949.

[14] Stanislas Dehaene. 1992. Varieties of numerical abilities. *Cognition* 44, 1 (1992), 1–42. DOI: http://dx.doi.org/https://doi.org/10.1016/0010-0277(92)90049-N

[15] Lynn S. Fuchs, Douglas Fuchs, Donald L. Compton, Sarah R. Powell, Pamela M. Seethaler, Andrea M. Capizzi, Christopher Schatschneider, and Jack M. Fletcher. 2006. The cognitive correlates of third-grade skill in arithmetic, algorithmic computation, and arithmetic word problems. *Journal of Educational Psychology* 98, 1 (2006), 29.

[16] Mark S. Handcock and Krista J. Gile. 2011. Comment: On the concept of snowball sampling. *Sociological Methodology* 41, 1 (2011), 367–371.

[17] Charles Hulme, Steven Roodenrys, Richard Schweickert, Gordon D. A. Brown, Sarah Martin, and George Stuart. 1997. Word-frequency effects on short-term memory tasks: Evidence for a redintegration process in immediate serial recall. *Journal of Experimental Psychology: Learning, Memory, and Cognition* 23, 5 (1997), 1217.

[18] Sana Inoue and Tetsuro Matsuzawa. 2007. Working memory of numerals in chimpanzees. *Current Biology* 17, 23 (2007). DOI:http://dx.doi.org/https://doi.org/10.1016/j.cub.2007.10.027

[19] J. H. Krantz. 2019. Psychological research on the net. (2019). https://psych.hanover.edu/research/exponnet.html

[20] Kevan Lee. 2014. Four Methods to Create a Secure Password You'll Actually Remember. (2014). `https://web.archive.org/web/20190123014846/https://www.lifehacker.com.au/2014/07/four-methods-to-create-a-secure-password-youll-actually-remember/` Accessed: 2017-12-18.

[21] Dirk Rijmenants. 2018. Hand Ciphers. (2018). `https://web.archive.org/web/20190610130334/http://users.telenet.be/d.rijmenants/en/handciphers.htm`

[22] Maja Rodic, Xinlin Zhou, Tatiana Tikhomirova, Wei Wei, Sergei Malykh, Victoria Ismatulina, Elena Sabirova, Yulia Davidova, Maria Grazia Tosto, Jean-Pascal Lemelin, and Yulia Kovas. 2015. Cross-cultural investigation into cognitive underpinnings of individual differences in early arithmetic. *Developmental Science* 18, 1 (2015), 165–174. DOI: `http://dx.doi.org/10.1111/desc.12204`

[23] Samira Samadi, Santosh Vempala, and Adam Tauman Kalai. 2018. Usability of Humanly Computable Passwords. In *6th AAAI Conference on Human Computation and Crowdsourcing*.

[24] Roger N. Shepard. 1967. Recognition memory for words, sentences, and pictures. *Journal of Verbal Learning and Verbal Behavior* 6 (02 1967), 156–163. DOI: `http://dx.doi.org/10.1016/S0022-5371(67)80067-7`

[25] Weining Yang, Ninghui Li, Omar Chowdhury, Aiping Xiong, and Robert W. Proctor. 2016. An Empirical Study of Mnemonic Sentence-based Password Generation Strategies. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS '16)*. ACM, New York, NY, USA, 1216–1229. DOI: `http://dx.doi.org/10.1145/2976749.2978346`