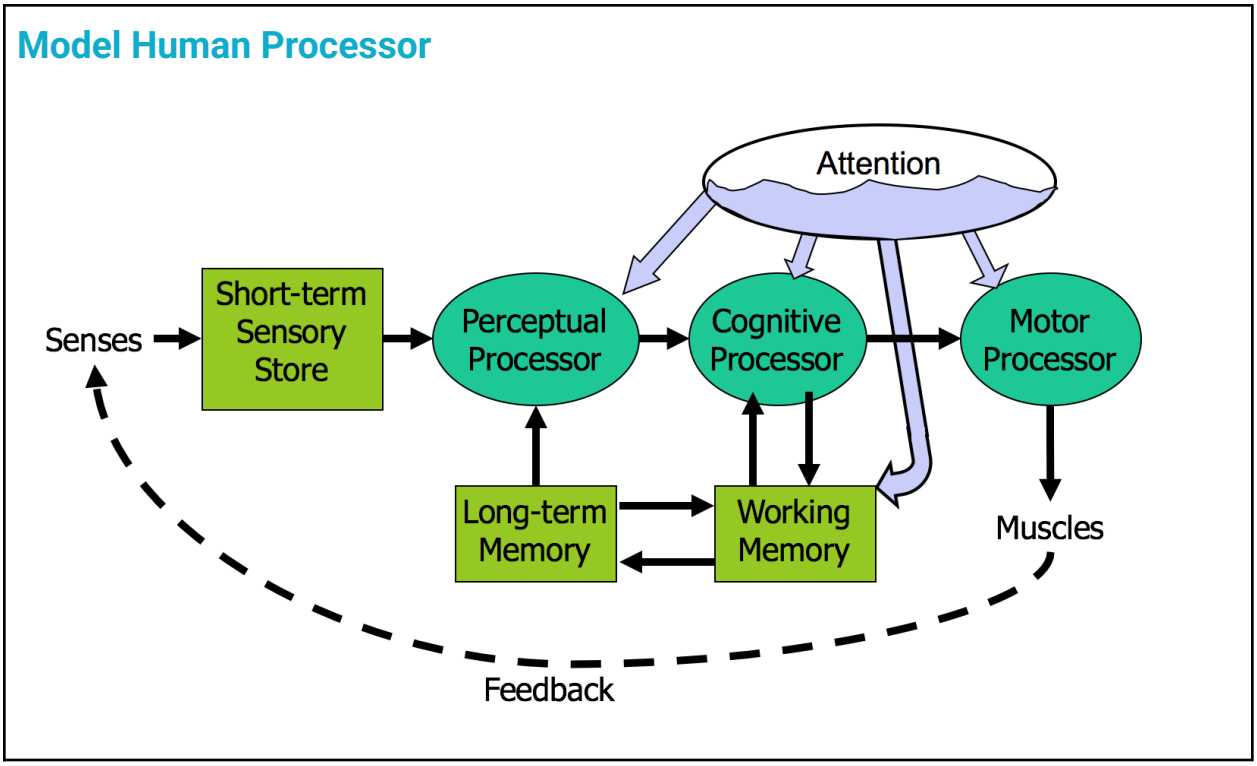
Human Abilities

Human Information Processing



Here's a high-level look at the cognitive abilities of a human being – really high level, like 30,000 feet. This is a version of the **Model Human Processor (MHP)**, which was developed by Card, Moran, and Newell as a way to summarize decades of psychology research in an engineering model (Card, Moran, Newell, *The Psychology of Human-Computer Interaction*, Lawrence Erlbaum Associates, 1983).

This model is different from the original MHP; we've modified it to include a component representing the human's attention resources (Wickens, *Engineering Psychology and Human Performance*, Charles E. Merrill Publishing Company, 1984).

This model is an abstraction, of course. But it's an abstraction that actually gives us numerical parameters describing how we behave. Just as a computer has memory and processor, so does our model of a human. Actually, the model has several different kinds of memory and several different processors.

Input from the eyes and ears is first stored in the **short-term sensory store**. As a computer hardware analogy, this memory is like a frame buffer, storing a single frame of perception.

The **perceptual processor** takes the stored sensory input and attempts to recognize symbols in it: letters, words, phonemes, icons. It is aided in this recognition by the **long-term memory**, which stores the symbols you know how to recognize.

The **cognitive processor** takes the symbols recognized by the perceptual processor and makes comparisons and decisions. It might also store and fetch symbols in **working memory** (which you might think of as RAM, although it's pretty small). The cognitive processor does most of the work that we think of as "thinking".

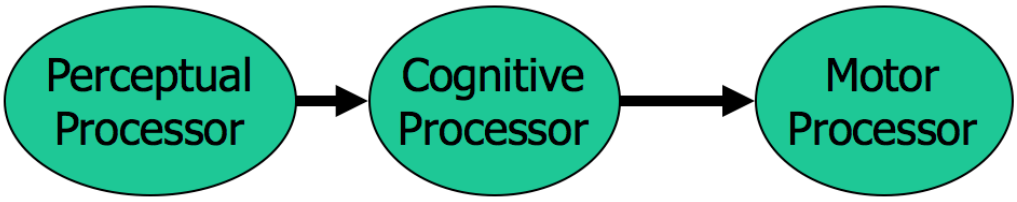
The **motor processor** receives an action from the cognitive processor and instructs the muscles to execute it. There's an implicit feedback loop here: the effect of the action (either on the position of your body or on the state of the world) can be observed by your senses, and used to correct the motion in a continuous process.

Finally, there is a component corresponding to your **attention**, which might be thought of like a thread of control in a computer system (coordinating multiple components for execution). Note that this model isn't meant to reflect the anatomy of your nervous system. There probably isn't a single area in your brain corresponding to the perceptual processor, for example. But it's a useful abstraction nevertheless.

* "A thread of control: a sequence of instructions being executed in a program" (ref)

Human Processor Cycle Time

- Processors have a cycle time
  - T_p ~ 100ms [50-200 ms]
  - T_c ~ 70ms [30-100 ms]
  - T_m ~ 70ms [25-170 ms]



The main property of a processor is its **cycle time**, which is analogous to the cycle time of a computer processor. It's the time needed to accept one input and produce one output.

Like all parameters in the MHP, the cycle times shown above are derived from a survey of psychological studies. Each parameter is specified with a typical value and a range of reported values. For example, the typical cycle time for perceptual processor, T_p, is 100 milliseconds, but various psychology studies over the past decades have reported mean cycle times between 50 and 200 milliseconds. The reason for the range is not only variance in individual humans; it also varies with conditions. For example, the perceptual processor is faster (shorter cycle time) for more intense stimuli, and slower for weak stimuli. You can't read as fast in the dark. Similarly, your cognitive processor actually works faster under load. Consider how fast your mind works when you're driving or playing a video game, relative to sitting quietly and reading. The cognitive processor is also faster on practiced tasks.

It's reasonable, when we're making engineering decisions, to deal with this uncertainty by using all three numbers, not only the nominal value but also the range.

Perceptual Fusion

- Two stimuli within the same PP cycle (T_p ~ 100ms) appear fused
  - Causality is strongly influenced by fusion
- Consequences
  - 1/ T_p frames/sec is enough to perceive a moving picture (10 fps OK, 20 fps smooth)
  - Computer response < T_p feels instantaneous
  - Causality is strongly influenced by fusion

One interesting effect of human perceptual system is **perceptual fusion**. Here's an intuition for how fusion works. Our perceptual processor runs at a certain frame rate, grabbing one frame (or picture) every cycle, where each cycle takes T_p seconds. Two events occurring less than the cycle time apart are likely to appear in the same frame. If the events are similar – e.g., Mickey Mouse appearing in one position, and then a short time later in another position – then the events tend to fuse into a single perceived event – a single Mickey Mouse, in motion.

Perceptual fusion is responsible for the way we perceive a sequence of movie frames as a moving picture, so the parameters of the perceptual processor give us a lower bound on the frame rate for believable animation. 10 frames per second is good enough for a typical case, but 20 frames per second is better for most users and most conditions.

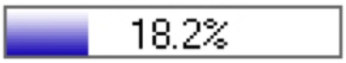
Perceptual fusion also gives an upper bound on good computer response time. If a computer responds to a user's action within T_p time, its response feels instantaneous with the action itself. Systems with that kind of response time tend to feel like extensions of the user's body. If you used a text editor that took longer than T_p response time to display each keystroke, you would notice.

Fusion also strongly affects our perception of causality. If one event is closely followed by another – e.g., pressing a key and seeing a change in the screen – and the interval separating the events is less than T_p, then we are more inclined to believe that the first event caused the second.

Response Time

- shorter than 0.1 s: seems instantaneous
- 0.1-1 s: user notices the delay
- 1-5 s: display a busy indicator

- longer than 5 s: display a progress bar



Perceptual fusion provides us with some rules of thumb for responsive feedback.

If the system can perform a command in less than 100 milliseconds, then it will seem instantaneous, or near enough. As long as the result of the command itself is clearly visible – e.g., in the user's locus of attention – then no additional feedback is required.

If it takes longer than the perceptual fusion interval, then the user will notice the delay – it won't seem instantaneous anymore. Something should change, visibly, within 100 ms,

or perceptual fusion will be disrupted. Normally, however, ordinary low-level feedback is enough to satisfy this requirement, such as a push-button popping back, or a menu disappearing.

One second is a typical turn-taking delay in human conversation – the maximum comfortable pause before you feel the need to fill the gap with something, even if it's just "uh" or "um". If the system's response will take longer than a second, then it should display additional feedback. For short delays, the hourglass cursor (or spinning cursor or throbber icon shown here) is a common design pattern. For longer delays, show a progress bar, and give the user the ability to cancel the command.

Note that progress bars don't necessarily have to be accurate. (18.2% is actually preposterous – who cares about 3 significant figures of progress?) An effective progress bar has to show that progress is being made, and allow the user to estimate completion time at least within an order of magnitude – a minute? 10 minutes? an hour? a day?

Cognitive Processing

- Cognitive processor
  - compares stimuli
  - selects a response
- Types of decision making
  - Skill-based
  - Rule-based
  - Knowledge-based

The cognitive processor is responsible for making comparisons and decisions. Cognition is a rich, complex process. The best-understood aspect of it is **skill-based** decision making. A skill is a procedure that has been learned thoroughly from practice; walking, talking, pointing, reading, driving, typing are skills most of us have learned well. Skill-based decisions are automatic responses that require little or no attention. Since skill-based decisions are very mechanical, they are easiest to describe in a mechanical model like the one we're discussing.

Two other kinds of decision making are **rule-based**, in which the human is consciously processing a set of rules of the form if X, then do Y; and **knowledge-based**, which involves much higher-level thinking and problem-solving.

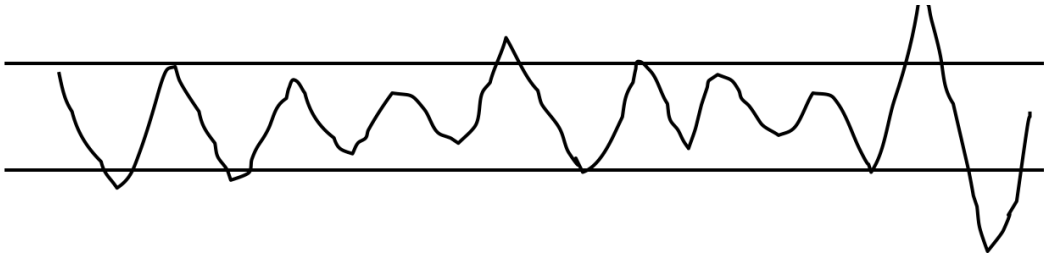
Rule-based decisions are typically made by novices or occasional performers of a task. When a student driver approaches an intersection, for example, they must think explicitly about what they need to do in response to each possible condition ("is there a stop sign? Are there other cars arriving at the intersection? Who has the right of way?"). With practice, the rules become skills, and you don't think about how to do them anymore.

Knowledge-based decision making is used to handle unfamiliar or unexpected problems, such as figuring out why your car won't start.

We'll focus on skill-based decision making for the purposes of this reading, because it's well understood, and because efficiency is most important for well-learned procedures.

Motor Processing

- Open-loop control
  - Motor processor runs a program by itself
  - cycle time is T_m ~ 70 ms
- Closed-loop control
  - Muscle movements (or their effect on the world) are perceived and compared with desired result
  - cycle time is T_p + T_c + T_m ~ 240 ms



The motor processor can operate in two ways. It can run autonomously, repeatedly issuing the same instructions to the muscles. This is "**open-loop**" control; the motor processor receives **no feedback from the perceptual system** about whether its instructions are correct. With open loop control, the maximum rate of operation is one cycle every T_m ~ 70 ms.

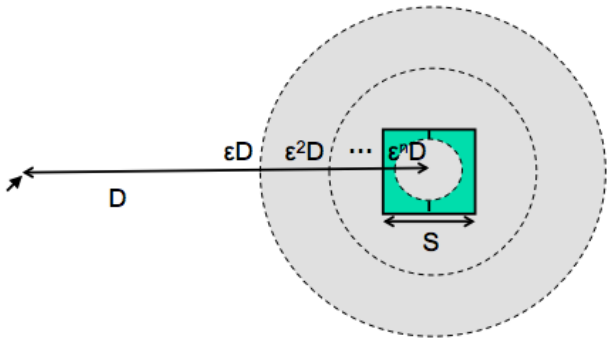
The other way is "**closed-loop**" control, which has a **complete feedback** loop. The perceptual system looks at what the motor processor did, and the cognitive system makes a decision about how to correct the movement, and then the motor system

issues a new instruction. At best, the feedback loop needs one cycle of each processor to run, or T_p + T_c + T_m ~ 240 ms.

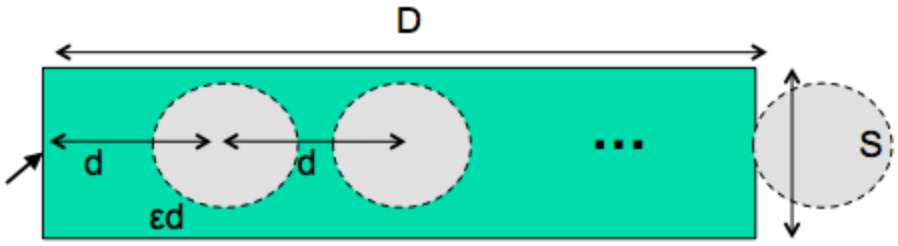
Here's a simple but interesting experiment that you can try: take a sheet of (thin) lined paper and scribble a sawtooth wave back and forth between two lines, going as fast as you can but trying to hit the lines exactly on every peak and trough. Do it for 5 seconds. The frequency of **the sawtooth carrier wave** is dictated by open-loop control, so you can use it to derive your T_m. The frequency of **the wave's envelope**, the corrections you had to make to get your scribble back to the lines, is closed-loop control. You can use that to derive your value of T_p + T_c.

Fitts's Law and Steering Law

- Fitts's Law
  - Time T to move your hand to a target of size S at distance D away is: T = RT + MT = a + b log (D/S + 1)
    - Moving your hand to a target is closed-loop control
    - Each cycle covers remaining distance D with error εD



- Steering Law
  - Time T to move your hand through a tunnel of length D and width S is: T = a + b D/S



We already appealed to this model of closed-loop motor control to help explain why Fitts's Law is logarithmic in distance and size, while the steering law is linear.

Choice Reaction Time

- Reaction time depends on information content of stimulus
  - RT = c + d log2 (1/Pr(stimulus))
- e.g., for N equiprobable stimuli, each requiring a different response:
  - RT = c + d log2 N

Simple reaction time – responding to a single stimulus with a single response – takes just one cycle of the human information processor, i.e. T_p+T_c+T_m.

But if the user must make a choice – choosing a different response for each stimulus – then the cognitive processor may have to do more work. The Hick-Hyman Law of Reaction Time shows that the number of cycles required by the cognitive processor is proportional to the amount of *information* in the stimulus. For example, if there are N equally probable stimuli, each requiring a different response, then the cognitive processor needs log N cycles to decide which stimulus was actually seen and respond appropriately. So if you double the number of possible stimuli, a human's reaction time only increases by a constant.

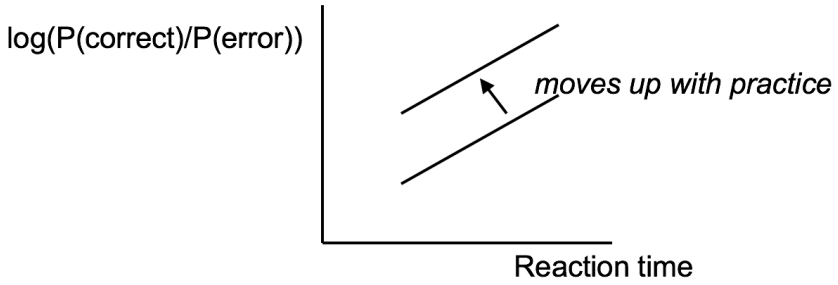
One intuitive explanation is that people subdivide the total collection of choices into categories, eliminating about half of the remaining choices at each step (just like a *binary search*!), rather than considering each and every choice one-by-one, which would

require linear time (a linear search). The *logarithmic form* O(log N) is due to this binary searching behavior.

Keep in mind that this law applies only to skill-based decision making; we assume that the user has practiced responding to the stimuli, and formed an internal model of the expected probability of the stimuli.

Speed-Accuracy Tradeoff

- Accuracy varies with reaction time
  - Here, accuracy is probability of slip or lapse
  - Can choose any point on curve
  - Can move curve with practice



Another important phenomenon of the cognitive processor is the fact that we can tune its performance to various points on a **speed-accuracy tradeoff** curve. We can force ourselves to make decisions faster (shorter reaction time) at the cost of making some of those decisions wrong. Conversely, we can slow down, take a longer time for each decision and improve accuracy. It turns out that for skill-based decision making, reaction time varies linearly with the log of odds of correctness; i.e., a constant increase in reaction time can double the odds of a correct decision.

The speed-accuracy curve isn't fixed; it can be moved up by practicing the task. Also, people have different curves for different tasks; a pro tennis player will have a high curve for tennis but a low one for surgery.

One consequence of this idea is that efficiency can be traded off against safety. Most users will seek a speed that keeps slips to a low level, but doesn't completely eliminate them.

Power Law of Practice

- Time T_n to do a task the nth time is:
  - $T_n = T_1 n^{-\alpha}$
    - α is typically 0.2-0.6

One more relevant feature of the entire perceptual-cognitive-motor system is that the time to do a task decreases with practice. In particular, the time decreases according to a power law. The power law describes a linear curve on a log-log scale of time and number of trials.

In practice, the power law means that novices get rapidly better at a task with practice, but then their performance levels off to nearly flat (although still slowly improving).

Locus of Attention

Feedback Visibility Depends on Locus of Attention

- **Spotlight of attention**: attention focuses on one input channel (e.g., area of visual field) at a time
- Does the user's locus of attention include:
  - Caps Lock light on keyboard?
  - Status bar?
  - Menu bar?
  - Mouse cursor?

The metaphor used by cognitive psychologists for how attention behaves in perception is the spotlight: you can focus your attention on only one input channel in your environment at a time. This input channel might be a location in your visual field, or it

might be a location or voice in your auditory field. You can shift your attention to another channel, but at the cost of giving up your previous focus.

So when you're thinking about how to make something important visible, you should think about where the user's attention is likely to be focused – their document? The text cursor? The animated banner ads on the web site?

Raskin, *The Humane Interface*, 2000, has a good discussion of attention as it relates to mode visibility. Raskin argues that we should think of it as the locus of attention, rather than focus, to emphasize that it's merely the place where the user's attention happens to be, and doesn't necessarily reflect any conscious focusing process on the user's part.

The status bar probably isn't often in the locus of attention. There's an amusing story (possibly urban legend) about a user study mainly involving ordinary spreadsheet editing tasks, in which every five minutes the status bar would display "There's a $50 bill taped under your chair. Take it!" In a full day of testing, more than a dozen users, nobody took the money. (Alan Cooper, *The Inmates Are Running the Asylum*.)

But there's also evidence that many users pay no attention to the status bar when they're considering whether to click on a hyperlink; in other words, the URL displayed in the status bar plays little or no role in the link's information scent. Phishing websites (fake websites that look like major sites like eBay or PayPal or CitiBank and try to steal passwords and account information) exploit this to hide their stinky links.

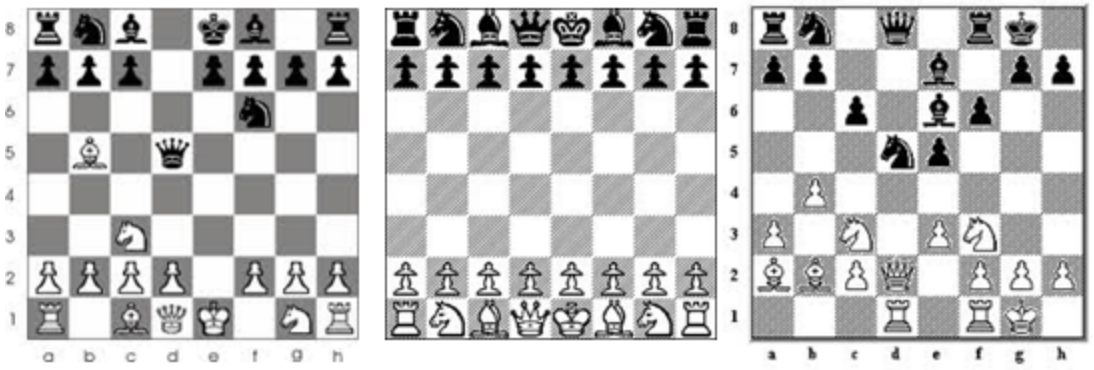
The Mac OS menubar has a similar problem – it's at the periphery of the screen, so it's likely to be far from the user's locus of attention. Since the menubar is the primary indicator of which application is currently active – in Mac OS, an application can be active even if it has no windows on the screen – users who use keyboard shortcuts heavily may make **mode errors** – in this case, sending a keyboard command to the wrong application – because they aren't attending to the state of the menubar.

What about the shape of the mouse cursor? Surely that's reliably in the user's locus of attention? It may be likely to be in the user's center of vision (or **fovea**), but that doesn't necessarily mean they're paying attention to the cursor, as opposed to the objects they're working with. Raskin describes a mode error he makes repeatedly with his favorite drawing program, despite the fact that the mode is clearly indicated by a different mouse cursor.

Chunking

Chunking

- "Chunk" is a unit of memory or perception
    - Depends both on presentation and on what you already know



The elements of perception and memory are called **chunks**. In one sense, chunks are defined symbols; in another sense, a chunk represents the activation of past experience. Chunking is illustrated well by a famous study of chess players. Novices and chess masters were asked to study chess board configurations and recreate them from memory. The novices could only remember the positions of a few pieces. Masters, on the other hand, could remember entire boards, but only when the pieces were arranged in *legal* configurations. When the pieces were arranged randomly, masters were no better than novices. The ability of a master to remember board configurations derives from their ability to chunk the board, recognizing patterns from their past experience of playing and studying games. (De Groot, A. D., *Thought and choice in chess*, 1965.)

Working Memory

- Small: 4 ± 1 "chunks"
- Short-lived: ~10 sec
- Maintenance rehearsal fends off decay (but costs attention)

Working memory is where you do your conscious thinking. The currently favored model in cognitive science holds that working memory is not actually a separate place in the brain, but rather a pattern of activation of elements in the long-term memory. A famous old result is that the capacity of working memory is roughly 7 ± 2 chunks. A recent reanalysis of the same experiment has amended that estimate to 4 ± 1 chunks (Parker,

“Acta is a four-letter word,” Acta Psychiatrica Scandinavica, 2012). Either way, it’s pretty small! Although working memory size can be increased by practice (if the user consciously applies mnemonic techniques that convert arbitrary data into more memorable chunks), it’s not a good idea to expect the user to do that. A good interface won’t put heavy demands on the user’s working memory.

Data put in working memory disappears in a short time–a few seconds or tens of seconds. Maintenance rehearsal—repeating the items to yourself—fends off this decay, but maintenance rehearsal requires attention. So distractions can easily destroy working memory.

Long-term memory is probably the least understood part of human cognition. It contains the mass of our memories. Its capacity is huge, and it exhibits little decay. Long-term memories are apparently not intentionally erased; they just become inaccessible.

Maintenance rehearsal (repetition) appears to be useless for moving information into long-term memory. Instead, the mechanism seems to be elaborative rehearsal, which seeks to make connections with existing chunks. Elaborative rehearsal lies behind the power of mnemonic techniques like associating things you need to remember with familiar places, like rooms in your childhood home. But these techniques take hard work and attention on the part of the user. One key to good learnability is making the connections as easy as possible to make–and consistency is a good way to do that.

Improve Efficiency of Output

- Present information in easily-recognized chunks

Hard: M W B C R A L O A B I M B F I
Easier: MWB / CRA / LOA / BIM / BFI
Easiest: BMW / RCA / AOL / IBM / FBI
Hard: 9405510200793831994315
Easier: 9405 / 5102 / 0079 / 3831 / 994 / 315
Easiest: klar / fonz / apek / uwer

Our ability to form chunks in working memory depends strongly on **how the information is presented**: a sequence of individual letters tend to be chunked as letters, but a sequence of three-letter groups tend to be chunked as groups. It also depends on what we already know. If the three letter groups are well-known TLAs (three-letter acronyms) with

well-established chunks in long-term memory, we are better able to retain them in working memory.

Take advantage of this as a designer: don't present information as long undifferentiated strings of random characters or numbers. At the very least, break them up into 3- or 4-character groups. Still better, find a way to make the chunks more familiar. This applies not just to random numbers or hashes, but to all kinds of data displayed in an interface.