"Hello, Dave. Shall we continue the game?" Questioning man-computer interactions

Liesbeth De Mol
Centre for Logic and Philosophy of Science, Belgium
Thanks to: Kunsthochschule für Medien, Köln
elizabeth.demol@ugent.be

Intro.

Introduction (1)

⇒ **Question** How to "talk" to a computer?

⇒ **Motivation** Omnipresence of computer + general lack of understanding of what goes on when one is "using" a computer

⇒ (Note: An *essay* – thinking in progress – from a humble computer "user" (historical, theoretical and everyday knowledge), **not** an HCI-expert)

Introduction (2)

- Two background assumptions/inspirations
- *An* approach: In search for opportunities/situations of man-computer "conversations"
- (history hard-and software) Lessons from the history of computing
- (foundations of computing) Unpredictability and the Church-Turing thesis
- (an application) Mathematical computer experiments as Man-computer "conversation"
- Discussion

Two background assumptions/inspirations

A1: The Heideggerian assumption/inspiration

Main idea: "Everywhere everything [also man] is ordered to stand by, to be immediately on hand" The danger of technique is that it is hidden away.

- Cecile Crutzen, 2000: "**Users are turned into resources which can be used by makers in the process of making IT-products. Users do not have room for starting their own designing processes. Those who do not fit in pregiven classes are seen as dissidents.**"
- Friedrick Kittler, 1987: "[I]f the ideal of software [...] would ever triumph, the bureaucratisation would be perfect: The hardware, in spite of its programmability, would irrevocably be obscured under its packaging. To stop this coincidence from happening seems to be an eminent political goal. **If computers are the first machines to reduce the contingency or incomputability of some, though not all futures to a finite degree, its own contingency should remain as open as possible.**"
- Edsger W. Dijkstra, 1985: "The point is that the computer user [...] is not a real person. [L]arge sections of computer science are paralyzed by accepting this moron as their typical customer **[U]ser friendliness is, among other things the cause of a frantic effort to hide the fact that eo ipso computers are mathematical machines**"

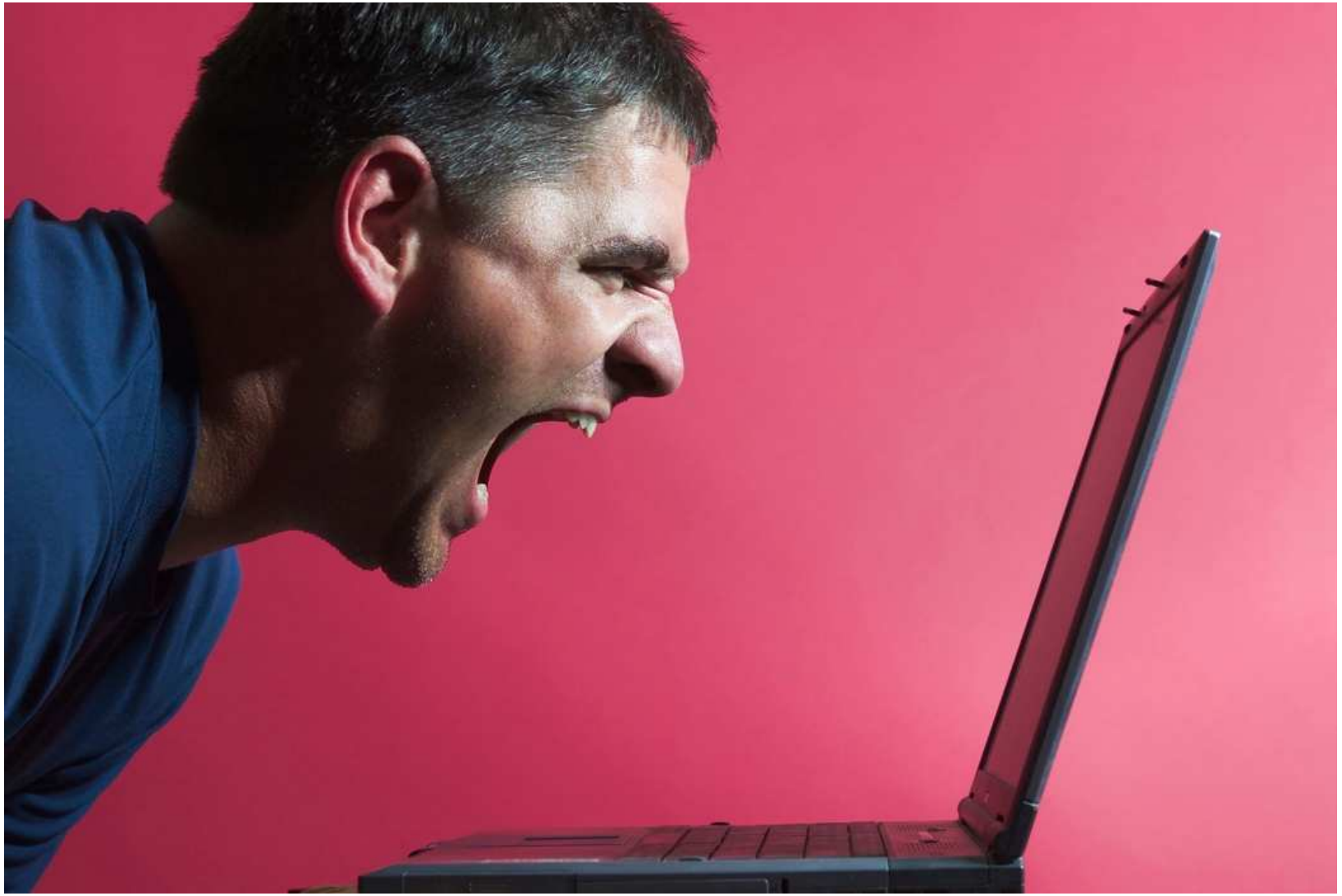
A2: The-fundamentally-different-assumption/inspiration

Main idea: interaction with something that is fundamentally different from us and allow it as such

- Licklider, 1960: "Computing machines can do readily, well, and rapidly many things that are difficult or impossible for man, and men can do readily and well, though not rapidly, many things that are difficult or impossible for computers. **That suggests that a symbiotic cooperation, if successful in integrating the positive characteristics of men and computers, would be of great value.**"

- Dijkstra, 1985: " Instead of trying to imitate what we are good at, I think it is much more fascinating to investigate what we are poor at. **It is foolish to use machines to imitate human beings, while machines are very good at being machines, and *that* is precisely something that human beings are very poor at.** Any successful AI project by its very nature would castrate the machine."

An approach: In search for opportunities/situations of man-computer "conversations"



An approach: In search for opportunities/situations of man-computer "conversations"

⇒ Man-computer "interactions" as "conversations"?

- allows focus on other "aspects" man-computer interaction that are characteristic for man-man-interactions/conversations
- (de-instrumentalization of the computer (A1): "So long as we represent technology as an instrument, we remain transfixed in the will to master it.")

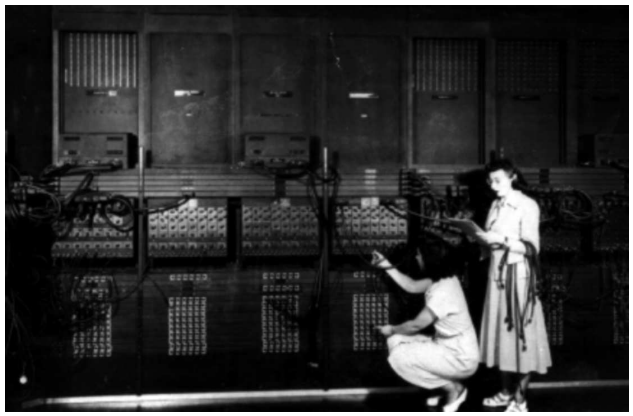
⇒ BUT: computer *as* computer, not a human (A2)

⇒ Transposition of "structural properties" man-man conversations to man-computer conversations: sensory contact, distance, common language, unpredictability and predictability, non-control, feedback, etc

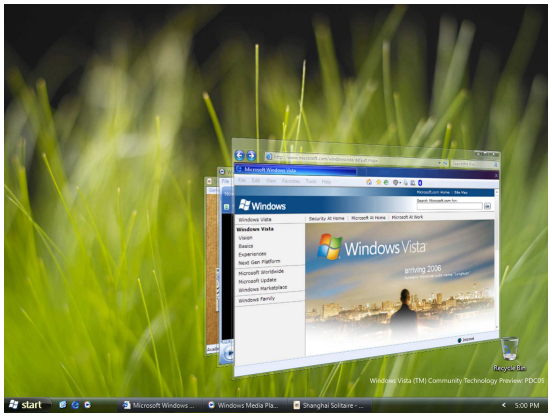
⇒ In search for opportunities/situations of man-computer "conversations" that are already there

- (History hard- and software) Lessons from the history of computing
- (Foundations of computing) Computability, Unpredictability and the Church-Turing thesis
- (Application) Computer experiments as "conversations"

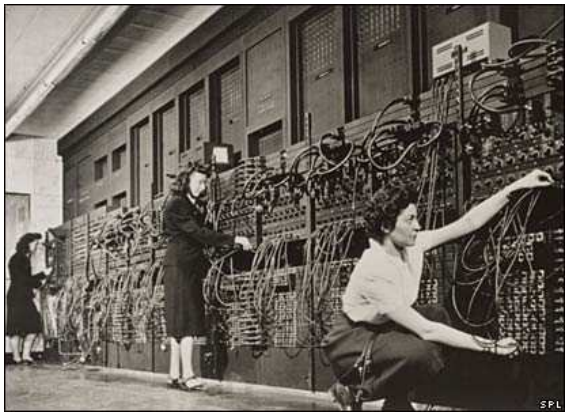
Lessons from the history of computing

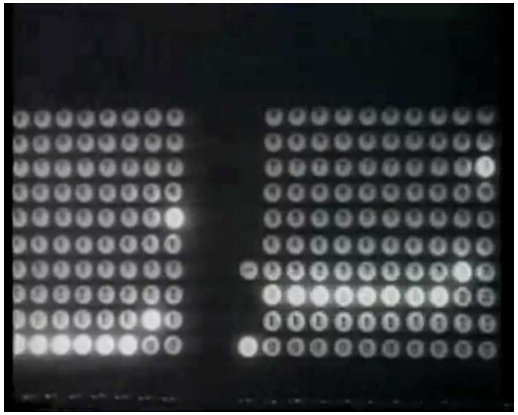


```
(define (initial-vec seq l n)
  (if (= l 0)
      seq
      (initial-vec (vector-append seq (vector (random n))) (- l 1) n)))
(initial-vec #(0) 300 2)


(define (add-period diff-periods n-period)
  (define (in-list? item seq)
    (cond ((null? seq) #f)
          ((equal? item (car seq)) #t)
          (else (in-list? item (cdr seq)))
          )
    )
  (cond ((= n-period (- 1)) diff-periods)
        ((in-list? n-period diff-periods) diff-periods)
        (else (append (list n-period) diff-periods))))

(add-period '(6) 1)

(define (determine-max r-max n-max?)
  (cond ((>= r-max n-max?) r-max)
        (else n-max?)))


(define (tag-it-vec count words v word ref-word ref-count)
  (define (deletion v word)
    (vector-tail word v))
  (define (produce words v word)
    (let ((appendent (vector-ref words (vector-first word))))
          (if (< (+ (- (vector-length word) v) (vector-length appendent)) 1) #()
              (deletion v (vector-append word appendent)))))
  (define (halt? word)
    (equal? word #()))
  (define (period? w1 w2)
    (equal? w1 w2))
  (define (period count ref-count)
    (- count ref-count))
  (let ((production (produce words v word)))
    (case (modulo count 1000000)
      ((0) (display count) (newline)(display (vector-length word)) (newline)))
    (cond ((period? word ref-word) (list count 0 1 (period count ref-count)))
          ((halt? production) (list count 1 0 (- 1)))
          ;((= count 100000) 1)
          ((= 0 (modulo count 1000)) (tag-it-vec (+ count 1) words v production word count))
          (else (tag-it-vec (+ count 1) words v production ref-word ref-count))
          )
    )
  )
```



Touching and sensing the (first) computer(s) (1)
How to "talk" with and "listen" to the behemoth ENIAC?





- "to talk": Programming through direct physical contact: the ENIAC "was a son-of-a-bitch to program" (De Mol & Bullynck, 2008)
- "to listen": Direct access to the computational process through sound and lights: "[W]hen you were doing calculations these lights were flashing as the numbers built up and as you transferred numbers and things of this kind. They were very essential to debugging, very essential. [...] That's the only way you read what the machine [...] stored, what it was doing. [I]t was [...] where people saw for the first time, saw calculations taking place" (Jean Bartik, 1973)

Touching and sensing the (first) computer(s) (2)
Problems with the "hands-on" approach

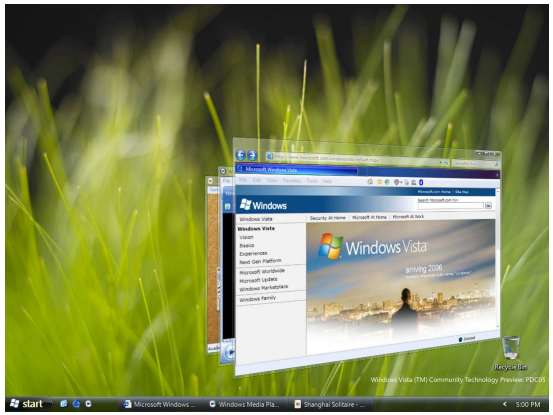
⇒ Completely non-efficient for both man and machine

⇒ Too close to the machine → feasible possibilities highly restricted → less "freedom" for both man and machine

⇒ More "responsibility" for the computer → Stored-program computer (the machine doing its own wiring) → development programming languages → Internalization of processes

Growing distances & internalization: "Programming" a computer (1)

```
(define (initial-vec seq i n)
  (if (= i 0)
      seq
      (initial-vec (vector-append seq (vector (random n))) (- i 1) n)))
(initial-vec #(3) 300 2)


(define (add-period diff-periods n-period)
  (define (in-list? item seq)
    (cond ((null? seq) #f)
          ((equal? item (car seq)) #t)
          (else (in-list? item (cdr seq)))
          )
    )
  (cond ((= n-period (- 1)) diff-periods)
        ((in-list? n-period diff-periods) diff-periods)
        (else (append (list n-period) diff-periods))))

(add-period '(6) 1)

(define (determine-max r-max n-max?)
  (cond ((>= r-max n-max?) r-max)
        (else n-max?)))


(define (tag-it-vec count words v word ref-word ref-count)
  (define (deletion v word)
    (vector-tail word v))
  (define (produce words v word)
    (let ((oppendent (vector-ref words (vector-first word))))
          (if (< (+ (- (vector-length word) v) (vector-length oppendent)) 1) #()
              (deletion v (vector-append word oppendent)))))
  (define (hait? word)
    (equal? word #(3)))
  (define (period? w1 w2)
    (equal? w1 w2))
  (define (period count ref-count)
    (- count ref-count))
  (let ((production (produce words v word)))
    (case (modulo count 1000000)
      ((0) (display count) (newline)(display (vector-length word)) (newline)))
    (cond ((period? word ref-word) (list count 0 1 (period count ref-count)))
          ((hait? production) (list count 1 0 (- 1)))
          ((= count 1000001) 1)
          ((= 0 (modulo count 1000)) (tag-it-vec (+ count 1) words v production word count))
          (else (tag-it-vec (+ count 1) words v production ref-word ref-count))
          )
    )
  )
```



- "Listening" and "talking" through a common language → interaction at some interface

⇒ man-computier interactions as "conversations"? → *conversation assumes distance*

Growing distances & internalization: "Programming" a computer (2)

⇒ In man-man conversation:

- no direct access to the others internal "processes"
- Mutual translations in conversation +making the "optimal" translation: taking into account the situation (the other; the dynamics and boundary conditions)

⇒ *Uncertainty* of mutual understanding;lack of total control → imposs. "perfect" or "error-free" communication

⇒ Bond in conversation; mutual need to understand each other, else master-slave reversal + "*we only know <u>what</u> we have said, when we have seen our listener reacted to it; we only know what the things we are going to say <u>will</u> mean in as far as we can predict his reaction. [...] As we do not master the behavior of the other, we badly need in speaking the feed back, known as "conversation".* (Dijkstra, 1961)"

⇒ What about man-computer "conversations"? Interface as condition to have man-computer "conversations", but also one that allows for this bond of mutual understanding ⇔ interfaces that are designed in function of the "moron" also known as the "user"; interface that hides what is going on beyond does not allow for man-computer "conversations"

Computability, Unpredictability and the Church-Turing thesis

Computability, Unpredictability and the Church-Turing thesis(1)

- Church-Turing thesis as a thesis at the root of what computing is:
  (rough version) *Anything that can be humanly computed can be computed by a Turing machine (or any other equivalent model)*
- Historical (and epistemological) significance: "For if symbolic logic has failed to give wings to mathematicians this study of symbolic logic opens up a new field concerned with the fundamental limitations of mathematics, more precisely the mathematics of Homo Sapiens" (Post in a letter to Church, May 26, 1936)
- "Indeed, with the bubble of symbolic logic as universal logical machine finally burst, a new future dawns for it as the indispensable means for revealing and developing those limitations." (Emil Post, 1943)

Computability, Unpredictability and the Church-Turing thesis (2)

⇒ The computer as *physical* and *finite* realization of computation & limits of human computation & speed computer

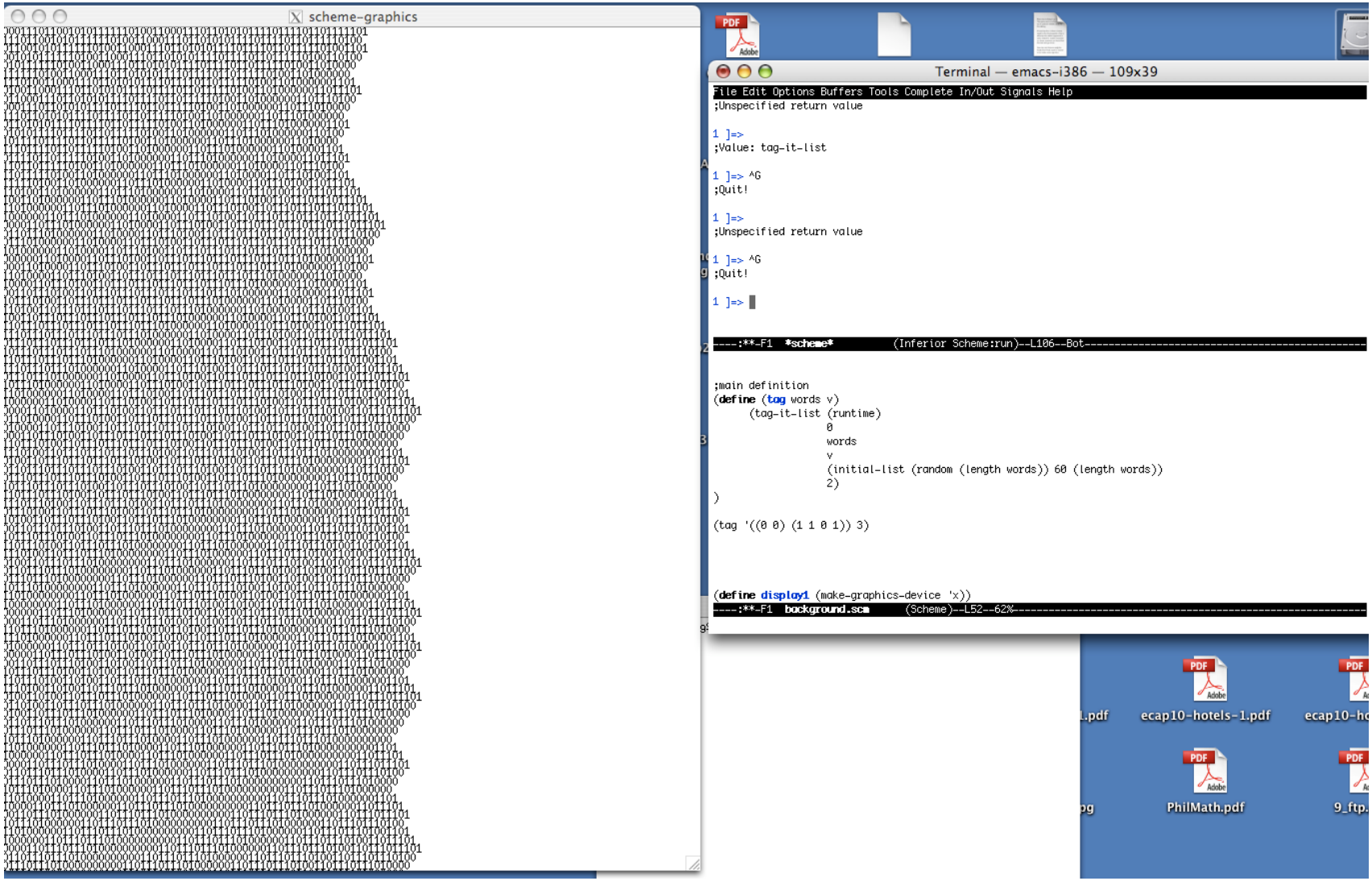
- Humanly impossible to predict the general behavior of the computer
- Impossibility of complete control (loads of examples! eg. Dow freefall of 7% in minutes;...)

⇒ The computer as a help to understand and explore mutual limitations and possibilities (think Von Neumann!)

⇒ Extra motivation to understand what a computation is (and thus to go back to the machine)

⇒ Accepting, allowing and exploring this unpredictability of computers and these limitations could result in more interesting conversations ∼ real man-man-conversations

Mathematical Computer Experiments as "conversations"



Mathematical Computer Experiments as "conversations"

⇒ *Computer experiments in math as examples of man-computer "conversations" as "collaborations"*

- "Poincaré anticipated the frustration of an important group of would-be computer users when he said, 'The question is not, What is the answer? The question is, What is the question?' One of the main aims of man-computer symbiosis is **to bring the computing machine effectively into the formulative parts of technical problems**. The other main aim is closely related. It is **to bring computing machines effectively into processes of thinking that must go on in "real time"**, time that moves too fast to permit using computers in conventional ways. [...]To think in interaction with a computer in the same way that you think with a colleague whose competence supplements your own will require much tighter coupling between man and machine" (Licklider, 1960)
- This is not the kind of machine proof with a "look, no hands!" point of view in which the machine starts from the postulates and proves a well know elementary theorem, *simulating* [m.i.] well-established heuristic procedures in its search for a proof. Rather it is **a man-machine cooperative endeavor**" (Lehmer, 1969)

Mathematical Computer Experiments as "conversations"

- An example: exploring the limits of computability through man-computer interactions with tag systems (De Mol, 2010)
- Experimentation through interface (internalization) $\Leftrightarrow$ experimenting with ENIAC
- Surprises and unpredictability! Without them, no need of computer in experimental research
- again: "we only know what we have said, when we have seen our listener reacted to it; we only know what the things we are going to say will mean in as far as we can predict his reaction. [...] As we do not master the behavior of the other, we badly need in speaking the feed back, known as "conversation"
- Not simply: I program, the computer gives answer, I have an hypothesis and then I test it – pysical aspects of computer + exploration, refinement of concepts, formulation of hypotheses, building up intuitions, etc – a process that bring both man and computer in the "thinking" process (possibility model human-machine interaction in experimentation?)

5. Discussion

Discussion

"*HAL: Dave, I don't know how else to put this, but it just happens to be an unalterable fact that I am incapable of being wrong.*"

- The lesson of HAL – "Illusion" of user-adapted "human" machine & demand of machine that follows orders (internalized infallibility)
- *Exploring* man-computer interactions as "conversations"?:
- Rethink the very notion of conversation in the context of man-computer conversations, avoiding to impose human or user-adapted behaviour on non-human
- Possibility of doing so rooted in what we already have

⇒ Significance of an understanding of what is going on beyond the GUI: Education!!

"*And they are fighting this machine, trying to get it to respond to their demands, finally succeeding; that's what a machine is to them. They really don't have any – I guess the way we say it today: they don't have a sense of identity with the machine.*" (D.H. Lehmer, 1972)

"*unless he loves his tools it is highly improbable that he will ever create something of superior quality*" (Dijkstra, 1962)