Introduction to GOMS Method

Introduction

- GOMS: Goals, Operators, Methods, and Selection Rules
  - Inspired by AI model (human problem solving)
  - Card, Moran, and Newel (1983)
  - Cognitive walkthrough modeling/design
    - Simulate a user's problem-solving process at each step through the dialogue, checking if the simulated user's goals and memory content can be assumed to lead to the next correct action
    - using a program to behave like a user in order to automatically test an interface, either for usability or quality assurance purposes → ACT-R
  - Alternative to empirical evaluation with users or experts

GOMS-KLM

- GOMS-KLM: Key-stoke Level Model
  - Pressing keys, moving the mouse, pressing buttons, … (called operators)
  - Task execution time is predicted by the total of times for the elementary keystroke-level actions to perform a task
  - Assumes an expert user

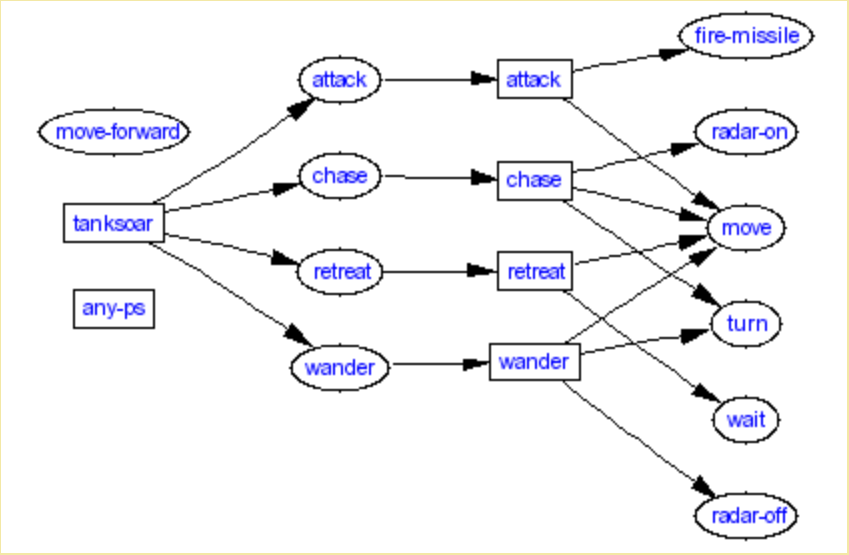
- GOMS models: descriptions of methods needed to accomplish specified goals
  - Methods have a hierarchical structure (call for sub-goals to be accomplished)
  - If more than one method is applicable at a step: selection rule chooses an appropriate method

GPS: General Problem Solving

Control

If (on A B), choose rule 1 over rule 2

…



Inference Engine (Rules)

Move B to table:

If (on table A) and (on A B)

Then (empty A) and (on table B)

Rule 1:

If menu-option-1 is selected

Then menu-level-2 activated

…

World (Facts in DB)

(on table cup)

(current-menu M1-1)

(selected left-mouse-button)

(satisfy-goal delete-file)

How to Model with KLM

- State assumptions

- Make a scenario
  - Write out action sequences and convert them to KLM-GOMS operators
  - Include wait operators if needed
  - Insert mental operators if needed

- Add things up to a total amount of time using KLM times estimates

Times for KLM Operators (Kieras,1993)

- K Keystroke
  - Expert: 0.12 sec
  - Average skilled typist: 0.20 sec
  - Average non secretarial: 0.28 sec
  - 독수리: 1.2 sec
- T(n): Type n characters 280*n msec
- P: Point with mouse to something on the display 1100 msec
- B: Press or release mouse button 100 msec
- BB: Click a mouse button 200 msec
- H: Home hands, either to the keyboard or mouse 400 msec
- M: “Mental”, thinking 1200 msec (can change)
- W(t): Waiting for the system to respond t msec (also called R time)

Writing a Model

- Task_item: T1
  - Name is First.
  - Type is delete_file.
  - Filename is "file1.txt".
  - Enclosing_directory is "Work".
  - Next is T2.

- Task_item: T2
  - Name is T2.
  - Type is move_file.
  - Filename is "file2.txt".
  - Enclosing_directory is "Work".
  - Destination_directory is "Documents".
  - Next is None.

Example in the paper

One file is to be deleted
File icon is visible and can be pointed to
Trash can icon is visible and can be pointed to
Cursor must end up in the original window that the file icon was in
Hand starts and ends on mouse
User is average non-secretary typist (40 wpm)

Action sequence

1. point to file icon
2. press and hold mouse button
3. drag file icon to trash can icon
4. release mouse button
5. point to original window

Operator sequence

1. point to file icon **P**
2. press and hold mouse button **B**
3. drag file icon to trash can icon **P**
4. release mouse button **B**
5. point to original window **P**

Total time = 3**P** + 2**B** = 3*1.1 + 2*.1 = 3.5 sec

Design 1

Action sequence

1. point to file icon
2. press and hold mouse button
3. drag file icon to trash can icon
4. release mouse button
5. point to original window

Operator sequence

1. point to file icon **P**
2. press and hold mouse button **B**
3. drag file icon to trash can icon **P**
4. release mouse button **B**
5. point to original window **P**

Total time = $3\mathbf{P} + 2\mathbf{B} = 3*1.1 + 2*.1 = 3.5$ sec

Design 2

Action sequence

1. point to file icon
2. click mouse button
3. point to file menu
4. press and hold mouse button
5. point to DELETE item
6. release mouse button
7. point to original window

Operator sequence

1. point to file icon **P**
2. click mouse button **BB**
3. point to file menu **P**
4. press and hold mouse button **B**
5. point to DELETE item **P**
6. release mouse button **B**
7. point to original window **P**

Total time = $4P + 4B = 4*1.1 + 4*.1 = 4.8$ sec

Design 3

- Use power key for delete
- Assume use both hands

Operator sequence

1. point to file icon **P**
2. click mouse button **BB**
3. move hand to keyboard **H**
4. hit command key command-T **KK**
5. move hand back to mouse **H**

Total time = **P** + 2**B** + 2**H** + 2**K** = 1.1 + .2 + .8 + .56 = 2.66 sec

Heuristics for Mental Operations

Initiating a task. The user has to pause and make a definite decision about what the task is, and what should be done. Thus users often pause before emitting a sequence of actions; this pause should be routinely represented by an M.

Making a strategy decision. If there is more than one way to proceed, and the decision is not obvious or well practiced, but is important, the user has to stop and think.

Retrieving a chunk from memory. A chunk is a familiar unit such as a file name, command name or abbreviation. For example, if the user wants to list the contents of directory `foo`, they need to retrieve two chunks, `dir` and `foo`, each of which takes an M.

Heuristics for Mental Operations

Finding something on the screen. The user must pause and scan the screen for an item that they do not already know or whose location on the screen they do not already know from practice.

Thinking of a task parameter. The user must either remember or otherwise access a task parameter value.

Verifying that a specification or action is correct. Before users signal the system to proceed, they often pause and check their entry. For example, users often stop and examine a command before hitting return, or check a dialog box before clicking on the OK, or that a destination folder is reverse-video before releasing the mouse button.

Some useful conventions. Models often contain certain sequences of actions which should be represented in a consistent way. Some suggestions:

- The values for task parameters have to be explicitly obtained in a step.
- Pointing to an object on the screen should be preceded by a mental operator to locate the object.
- If something on the screen changes in response to user input, there should be a step to veryify that the desired result appeared.

Heuristics for Mental Operations

New users verify every step. New users will stop and check feedback from system at every step, taking an **M** to do so. Experienced users skip the verification step, sometimes making errors as a result. There is some empirical evidence for this difference.

New users have small chunks, experienced users have big chunks. New users have to think about assembling commands from the basic pieces; they've not yet learned the larger patterns. Experienced users may have much more elaborate chunks, in which a whole complex command is zipped out as a single stream. For example, with the old MTS e-mail, I always used the editor to neaten up my message with `justify /file left 1 65`. This command was always the same, and I did it a lot, so it became for me a single chunk, typed in as a unit.

Experienced users can overlap Ms with physical operators. An apparent characteristic of highly practiced performance is the ability to do more than one thing at a time if it is physically possible. For example, a practiced user might be able to visually locate an icon on the screen while homing the hand to the mouse and starting the mouse movement. In this case, you can want to drop the **M** for finding the icon visually, leaving just the physical **H** and **P** operators.

Heuristics for Mental Operations

- Insert M's in front of all K's or B's that are not part of argument strings proper (e.g., text or numbers)
  - An argument is something that varies. For instance, if you burn CDs to files frequently, the files you burn are arguments. If you change fonts frequently, then the fonts are arguments.
  - If, however, you always change a font to a particular font, say FreeMono, then you might find that the font is no longer an argument but part of the command – the user might not think of changing fonts, but FreeMono-izing a section of text. Since users process arguments and commands differently, this can make a difference.

Chunking!

Heuristics for Mental Operations

- Place M's in front of all P's that select commands or that begin a sequence of direct-manipulation operations belonging to a cognitive unit.
  - Direct-manipulation operations are when you issue commands by manipulating things on-screen. For example, rather than selecting something like Files Delete, you drag something to a garbage can.

- If an operator following an M is fully anticipated in an operator just previous to M, then delete the M (e.g., PMK becomes PK or PMBB becomes PBB).
  - Fully anticipated means that, for instance here, the time it takes to perform the P operator (1100 msec) is almost as long as the time it takes to perform the M operator (1200 msec).

Heuristics for Mental Operations

- It is assumed that the user is executing the P operator while preparing for the next action.
- It is also the case that the "M" drops out because the P and BB belong together in a chunk.

- If a string of MK's or MB's belongs to a cognitive unit (e.g., the name of a command), then delete all M's but the first.
  - For instance, an emacs command like C-x 4 C-f (find-file-other-window) would be MKKK, rather than MKMKMK, because the command name, from the view of the user, is one sequence of three keystrokes.

Heuristics for Mental Operations

- If a K is a redundant terminator (e.g., the terminator of a command immediately following the terminator of its argument), then delete the M in front of it.

- If a K terminates a constant string (e.g., a command name), then delete the M in front of it; but if the K terminates a variable string (e.g., an argument string), then keep the M in front of it.

Example **with M's:**

Burning CDs with ECLiPt Roaster vs. GNOME Toaster

- Assumptions
  - We are starting from the startup screen of each program.
  - The user's hands are on the mouse.
  - A fresh CD is in the system, so the user doesn't need to insert one.
  - Use the Kieras operators.

Action Sequence for ECLiPt Roaster

1. Click Add.
2. Click slider down twice. Double click writing.
3. Click journal.
4. Click OK (adds journal to the list of things to burn).
   Click Close (closes the file dialog).
5. Click Burn CD. A dialog pops up, asking if, since you didn't select any files, you would like to burn all of them. Click Yes.
6. Another window opens up asking some more information about the disc (write speed, CD-R or CD-RW media, whether to fixate after burning). Click Burn

Final Operator Sequence

• M	selecting command	1200 msec
• P	point to Add	1100 msec
• BB	click	200 msec
• P	point to slider	1100 msec
• BB	click	200 msec
• BB	click	200 msec
• P	move to writing	1100 msec
• BBBB	double-click	400 msec
• P	move to journal	1100 msec
• BB	click	200 msec
• P	move to OK	1100 msec
• BB	click	200 msec

Final Operator Sequence

• M	selecting command	1200 msec
• P	move to Close	1100 msec
• BB	click	200 msec
• M	selecting command	1200 msec
• P	move to Burn CD	1100 msec
• BB	click	200 msec
• P	move to Yes	1100 msec
• BB	click	200 msec
• M	selecting command	1200 msec
• P	move to Burn	1100 msec
• BB	click	200 msec

sum: 16.9 seconds

Action Sequence for X-CD-Roast

- Click Create CD.
- Click Master Tracks.
- Click on the slider bar to make it go down twice.
- Click plus sign next to writing to make the directory open up.
- Click down on the slider one more time.
- Drag journal to the left-hand side of the screen.
- Select “Add to root directory” (of the CDROM).
- Click OK.
- Click Create session/image. This tab has settings on how long the CD-R is, whether to eject after write, and so forth.
- Click Master and write on-the-fly.

Final Operator Sequence

• M	selecting command	1200 msec
• P	point to Create CD	1100 msec
• BB	click	200 msec
• M	selecting command	1200 msec
• P	point to Master tracks	1100 msec
• BB	click	200 msec
• P	move to slider	1100 msec
• BB	click	200 msec
• BB	click	200 msec
• P	move to "+" next to writing	1100 msec
• BB	click	200 msec
• P	move to slider	1100 msec
• BB	click	200 msec
• P	move to journal	1100 msec
• B	mouse button down, to begin dragging	100 msec
• M	beginning a sequence of direct-manipulation operations	1200 msec

Final Operator Sequence

• P	drag to CD side of the screen	1100 msec
• B	mouse button up, to end dragging	100 msec
• M	selecting command	1200 msec
• P	point to “Add to root directory of CD”	1100 msec
• BB	click	200 msec
• M	before a B that is not part of an arg. string	1200 msec
• P	point to OK	1100 msec
• BB	click	200 msec
• M	selecting command	1200 msec
• P	move to Create session/image	1100 msec
• BB	click	200 msec
• M	selecting command	1200 msec
• P	move to Master and write on-the-fly	1100 msec
• BB	click	200 msec
• P	move to OK	1100 msec
• BB	click	200 msec

sum: 24 seconds

Observations

- In general,
  - Keystrokes are fast
    - But only if commands can be remembered quickly
    - Ease of learning and M's are related
  - M's are slow
  - Mouse is slow
    - No thinking but need to move and search
  - Switching between hands is slow

KLM – Evaluation

- Especially suitable for comparing efficiency of use of different systems, strategies

- Just descriptors; do not address problem solving (i.e. how to design)

- Other GOMS Models
  - NGOMSL (Kieras): Natural GOMS Language
  - CPM-GOMS (Gray, John, Atwood): Mapping of sequential dependencies between users' perceptual, cognitive and motor processes in an execution