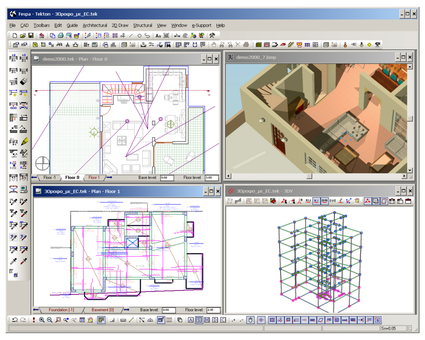
Echoes From Space:
Grouping Commands with Large-Scale Telemetry Data

Alexander Lattas
Imperial Colege London
alexandros.lattas17@ic.ac.uk
@alexanderlattas

Diomidis Spinellis
Athens University of Economics and Business
dds@aueb.gr
@CoolSWEng

The Command Grouping Problem

Evolving desktop applications continuously accrue new features and grow denser user interfaces with deeply-nested commands. Existing ***search-based software engineering*** studies on user performance prediction and command grouping optimization lack evidence-based answers on choosing a systematic grouping method.



Command Grouping: Systematic method for grouping commands into multi-level menus and toolbars for:
- ✔ End-user usability and efficiency
- ✔ Improved GUI design process

Model and Data

Based on the KLM model [1] for user perfomance time, in our ***model*** we additionally differentiate between the time needed when clicking commands in the currently open menu (*Same*), in a currently hidden nested menu (*Different*) and in an always open menu (*Toolbar*).

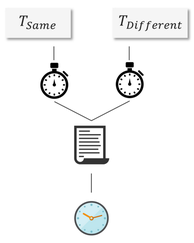
$$T_{Total} = T_{Same} \times (N_{Same} + N_{Toolbar}) + T_{Different} \times N_{Different}$$

With ***telemetry data*** of CAD program runs, we find real usage scenarios, with which we evaluate our models:
- → 32 million commands,
- → 200 thousand sessions,
- → 2000 users.

With controlled ***experiments***, we calculate T_{Same} and $T_{Different}$, with which the scenarios are objectively evaluated for each optimization algorithm.

Optimization Algorithms

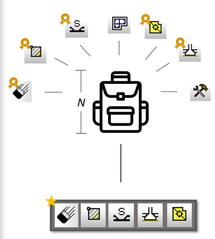
`CURRENT`:
Measurement with the GUI at the current setup.



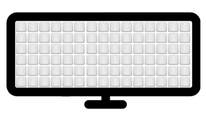
`NAIVE`:
N most frequent commands always available on an extra toolbar.

`MRU-B`:
N most frequent commands always available on a toolbar. For each user, from a small batch.
`MRU-O`:
For each user, online.

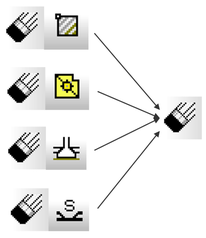
`OPT(KS)`:
N best commands always available on a toolbar, using a heuristic approximation of the Knapsack algorithm.



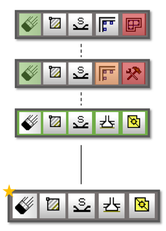
`ALL`:
All commands available directly on the screen. Establishes the theoretical optimum.



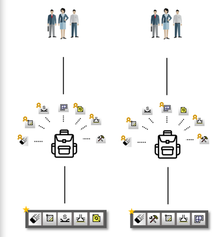
`GROUP`:
Group together similar commands between entities.



`OPT(GA)`:
N best commands always available on a toolbar, using a stochastic Genetic Algorithm.



`CLUSTER`:
Make clusters (kmeans) of users. Run OPT(KS) for each cluster. Show best N commands to each.

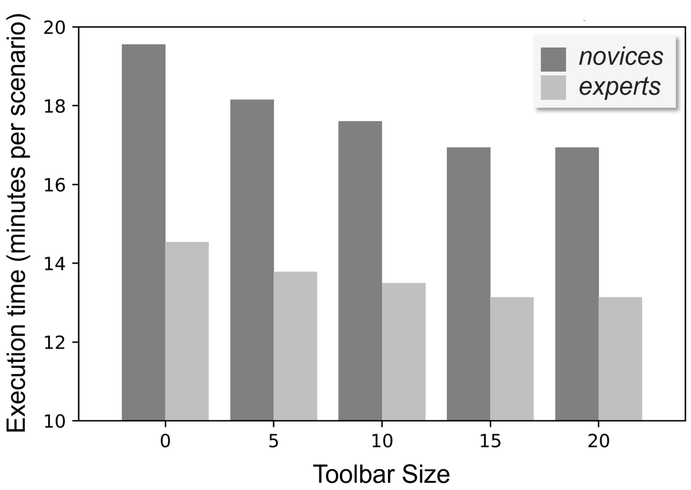


Results

When introducing an ***extra toolbar***, more than 15 of the best (`NAIVE`, `OPT(KS)`, `CLUSTER`) commands add no value.

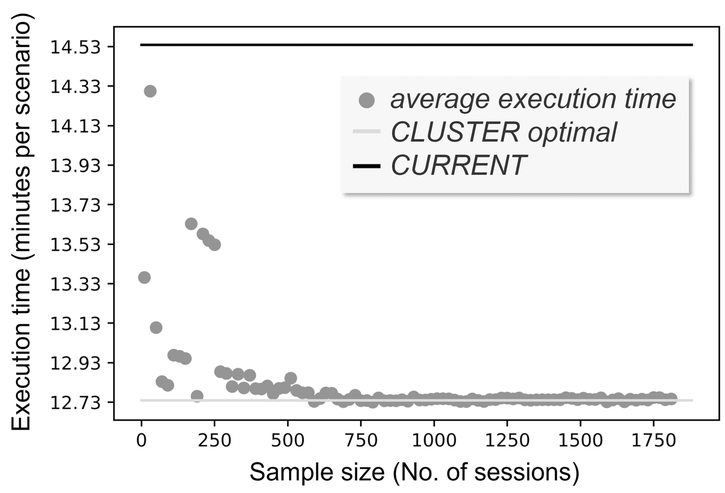
`OPT(KS)` and `CLUSTER` achieve optimal results. `GROUP` fails and `OPT(GA)` needs much training. `NAIVE` and `MRUs` good for quick solutions.

`CLUSTER` needs about 500 user sessions to optimize, `NAIVE` and `OPT(KS)` need about 2000, making **`CLUSTER`** ***the best algorithm.***



Method	Novices	Training Time (s)
ALL	18.5%	-
GROUP	0.6%	-
NAIVE	13.2%	7.8
MRU-B	10.4%	74.2
MRU-O	13.5%	1.2*
OPT(KS)	17.43%	1,946
OPT(GA)	17.40%	19,749*
CLUSTER	17.43%	2035

good
optimal
bad



References:
[1] Stuart K. Card, Thomas P. Moran, and Allen Newell. 1980. The Keystroke-Level Model for user performance time with interactive systems. *Commun. ACM 23,* 7 (1980), 396-410.

We thank LH Logismiki for providing us access to the system's source code and anonymized data. The project associated with this work has received funding from the EU's Horizion research and innovation programme 2020 (732223). The first author is a recipient of the Hellenic Petroleum Group scholarship.