Assignment 7: Design Optimization with Random Search

Jaume Benseny 463922

1. List non-trivial design decisions that are important for you at this point of the project.

- How many categories (layers) should I display on the interface?
- How many subcategories (within layers) should I display on the interface?
- What height should the layers? Should height change when categories are selected?

2. List all predictive models that you have access to, have implemented yourself, or could implement yourself now.

- KLM to estimate execution time of main user tasks (Modelling 1 exercise).
- SDP to optimize layout design based on design decisions (Modelling 2 exercise).

3. Determine the best matching decision & model combination.

- The best combination model would be to **first optimize layout** design based on SDP.
- Secondly **minimize query construction time** based on KLM model.
- **SUMMARY: Optimization minimizes KLM estimated time using SDP selection times.**

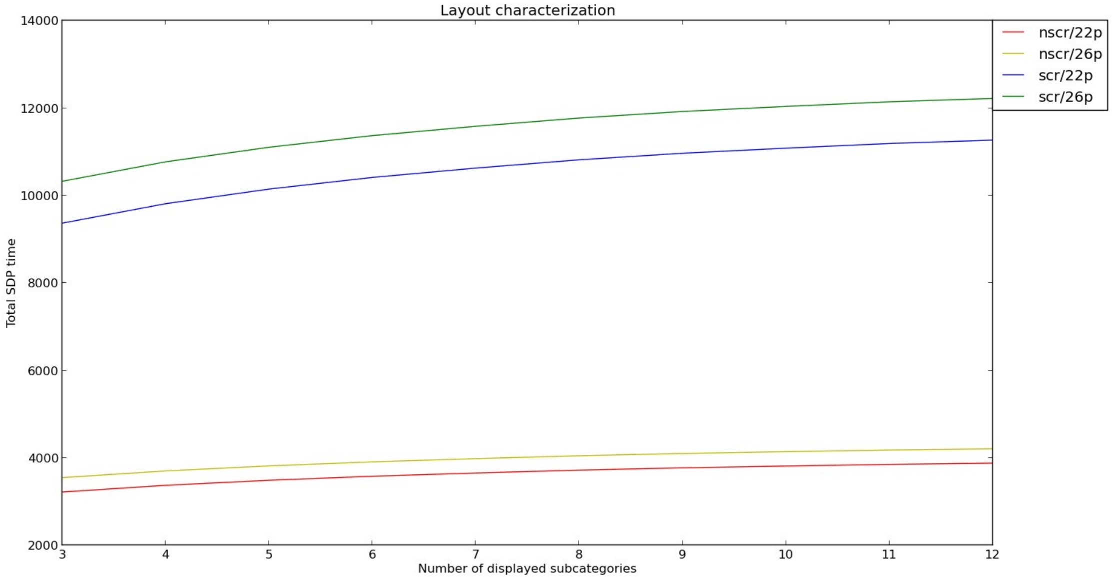
4. Define the design space determined by your design space. Ideally it should have more than 100 candidate designs. You may need to discretize variables etc. to make the space of feasible size.

Variables within the SDP model – Layout design

- Total number of categories and their **height** relative to **screen size**
- Number of subcategories / category their **width** relative to **screen size**
- Scrolling
- Item height (zoom in / amplification?)

Characterization of the present model as an update of last week exercise.
(next page)

- The following diagram describes SDP model adjusted to this interface:



- Compared to last week model, swipe time has been eliminated when selecting subcategories that are located out of the interface. In this case, the diagram compares estimated times for a and b parameters for scrolling and non-scrolling experiments.
- CONCLUSION 1. Scrolling require longer times to get to targeted menu item.
- **CONCLUSION 2. The larger the items are within the menu, the longer the user will take to find it. (counter-intuitive)**

Variables within the KLM model - Search complexity

- Probability of error (p_error) and “time to make it right” (t_error)
  - *t = (1 – p_error)* t_ok + p_error * t_error*
- **KLM TASK selected → selection of 2 categories + 2 subcategories per category**
  - *t_ok* = 2* *(tm* + *t_categories* + 2**t_subcategories* + *t_next* + 4**tr)*
- Probability of selection of categories and subcategories arbitrarily selected (pc1,pc2,psc1,psc2)

5. Implement evaluate() function

- **Evaluate function calculates total query construction time as:**
    1. **KLM estimated time of TASK**
    2. **Taking into account SDP selection times for the interface.**

- Execution calls the following functions:
    - def KLM_executiontime (categories, subcategories ,p_error, scrolling)
    - which calls as function:
    - def SDP_selectiontime (scrolling,menulength, itemposition, trials, p_i,item_height)

- Evaluate function cumulates optimal time as minimum time achieved for each query.

6. Implement random search optimizer

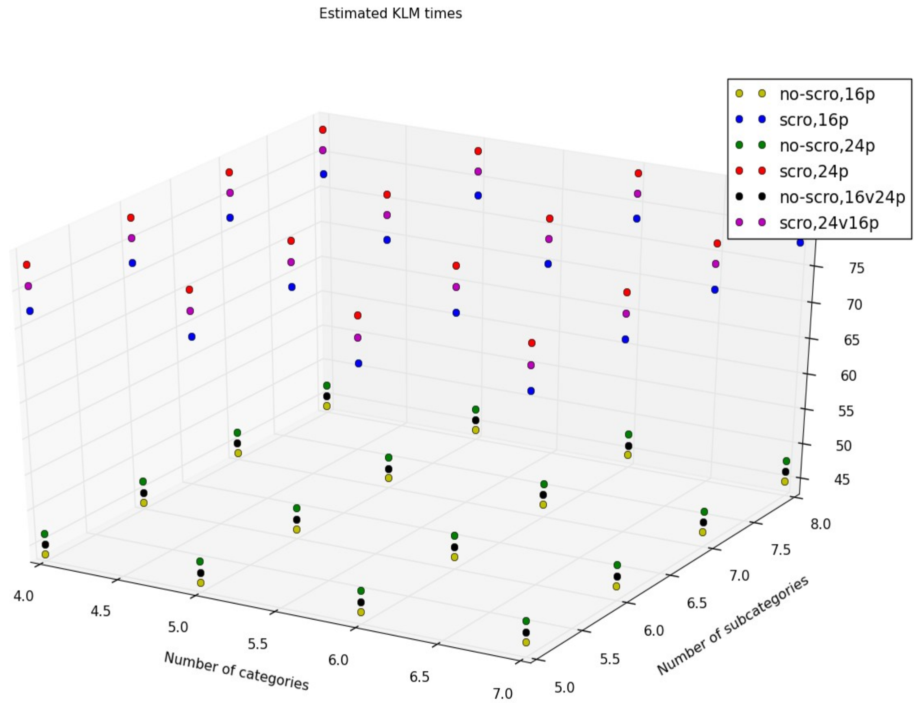
- Random search optimizer feeds the model with the following values:
    - categories € [4 – 6]
    - subcategories € [5 – 8]
    - scrolling € [0,1]
    - p_error constant to 0.3
    - height of layers that contain categories and subcategories
    - hc1 = hc 2 = 0.18
    - hc1 = 0.22, hc2 = 0.18
    - hc2 = 0.22, hc1 = 0.18

NOTE:

I tried for 2 hours to implement combinatorial optimization using NumberJack library for python and Mistral solver (Barry Hurley slides). I had to surrender after experiencing problems with python and Ubuntu package manager and import tools. After adding python-dev package from Ubuntu to get Mistral, still I coudn't import Mistral as Solver.

Random search optimizer has been implemented as a collection of loops that cross compared multiple results of the models.

- Optimal layout design based on this optimization process is:
  - optimal time 43.4811052235172
  - categories = 4
  - subcategories = 5
  - scrolling = 0
  - hc1 = hc 2 = 0.18



7. Conclusions

- The actual model and optimization strategy doesn't provide guidance towards advantages resulting from available design decisions.
- Behavior of the model is lineal and predictable.
- One result that is consistent with SDP model is that query construction time decreases when the height of category layer decreases.

8. Further development

- Implementation of combinatorial optimization with NumberJack framework.
- Introduction of constraints that relate screen size to the relative height of layers.
- Introduction of user behavioral decisions based on sales statistics.