SBSelector:
Search Based Component Selection for Budget Hardware

Lingbo Li, Mark Harman, Fan Wu, and Yuanyuan Zhang
CREST, UCL




UCL CREST

Introduction

SBSelector: Component Selection

Research Questions

Experiments and Results

Conclusions

Introduction

SBSelector: Component Selection

Research Questions

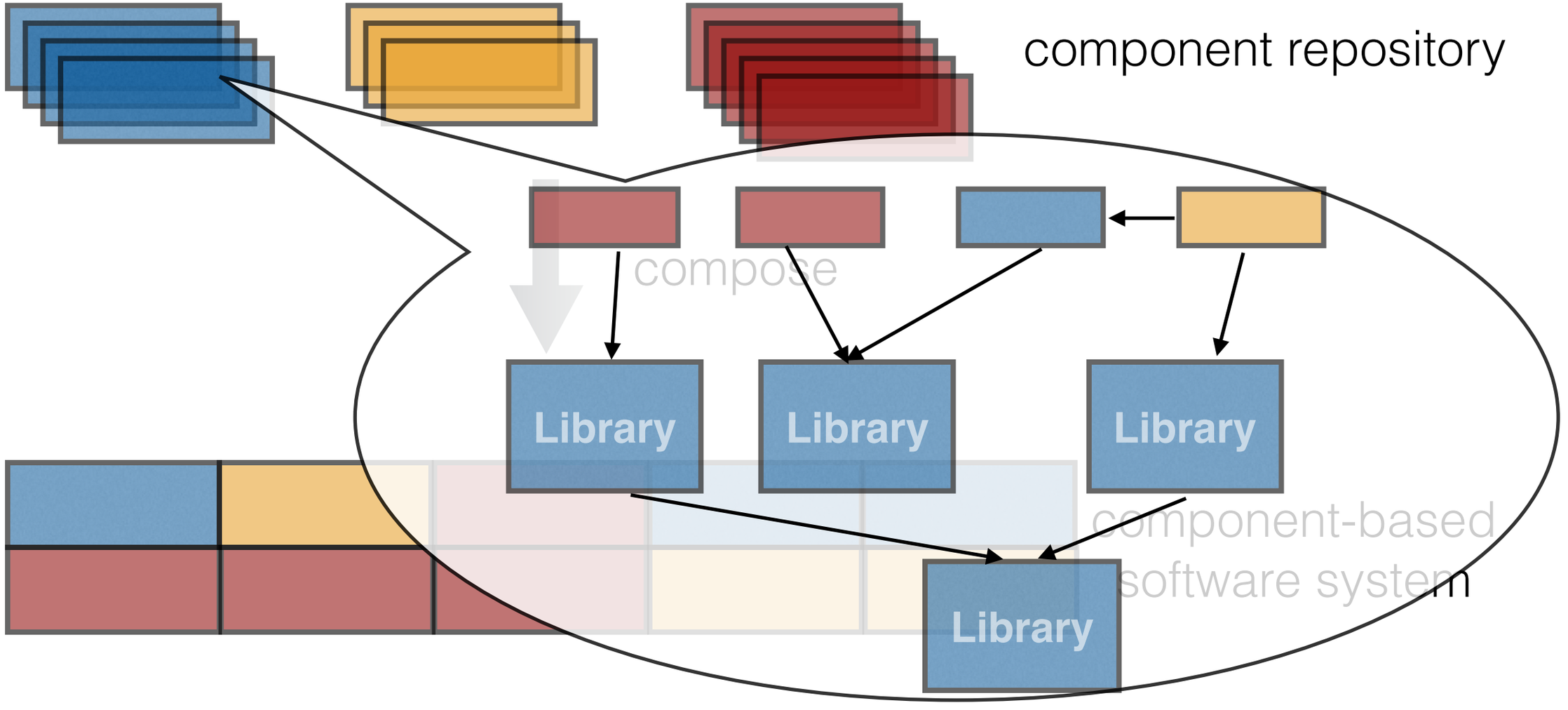
Experiments and Results

Conclusions

Introduction

Component based Software Engineering

Defining, implementing and composing loosely coupled independent components into systems



Select component: iteratively select desired component.

laborious

tricky

Introduction

SBSelector: Component Selection

Research Questions

Experiments and Results

Conclusions

SBSelector: Component Selection

SBSelector: a search based component selector.

Formulate component selection as a search problem.

Construct dependencies between components.

- precedence

Objective one: number of components (plugins)

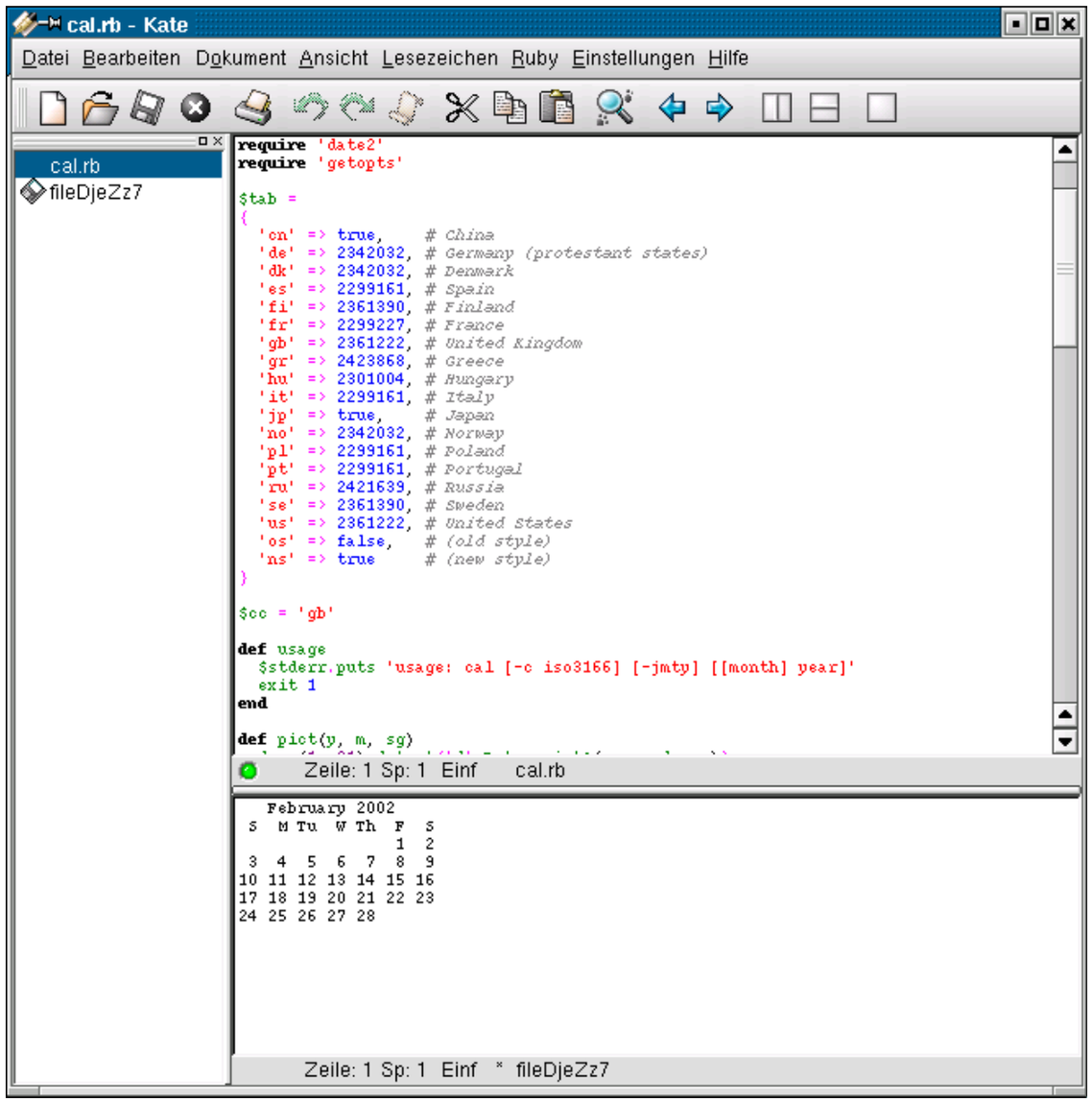
$$Maximize\ component(\vec{x}) = \sum_{i=1}^{n}(x_i)$$

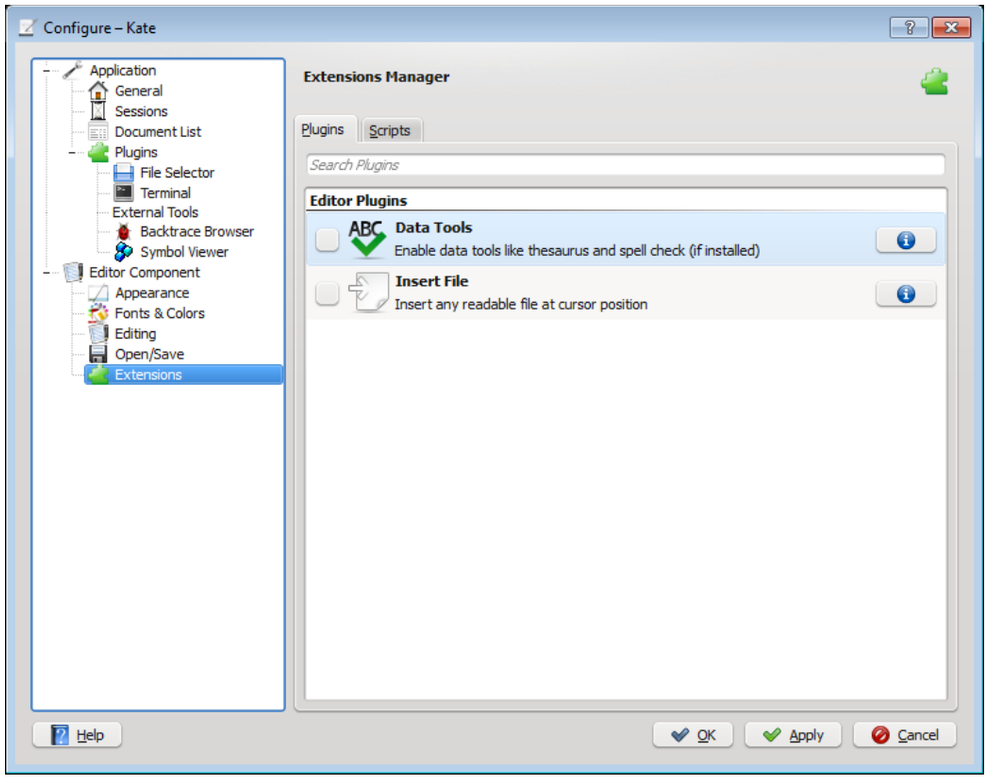
Objective two: worst-case memory consumption

$$Minimise\ memory(\vec{x})$$

Subject

Kate is a C/C++ based multi-platform text editor with component (plug-in) support.





Introduction

SBSelector: Component Selection

Research Questions

Experiments and Results

Conclusions

Research Questions

RQ1. Does the extra memory consumed by enabling all plugins of Kate simply equals to the summation of the extra memories consumed by enabling one of those plugins at a time?

$$\Delta(\vec{x_i}) = memory(\vec{x_i}) - memory(\vec{x_\emptyset})$$

$$\sum_i \Delta\vec{x}_i \stackrel{?}{=} \Delta\vec{x}_{all}$$

Research Questions

RQ2. How effectively can SBSelector find optimised combination of enabled plugins compared to random search method and a greedy strategy selection?

Assuming human developers (or users) may include components randomly or greedily based on the memory consumed.

Research Questions

RQ3. Given some mandatory plugins, can SBSelector still find combinations of optional plugins that only trade a little amount of memory consumption?

Scenario S1: all Python plugins, `Search and Replace', and `SQL Plugin' are essential for Python developers.

Scenario S2: all components are open to select.

Introduction

SBSelector: Component Selection

Research Questions

Experiments and Results

Conclusions

Answer to RQ1

RQ1.

$$\sum_i \Delta\vec{x}_i \approx 45.78\text{MB}$$

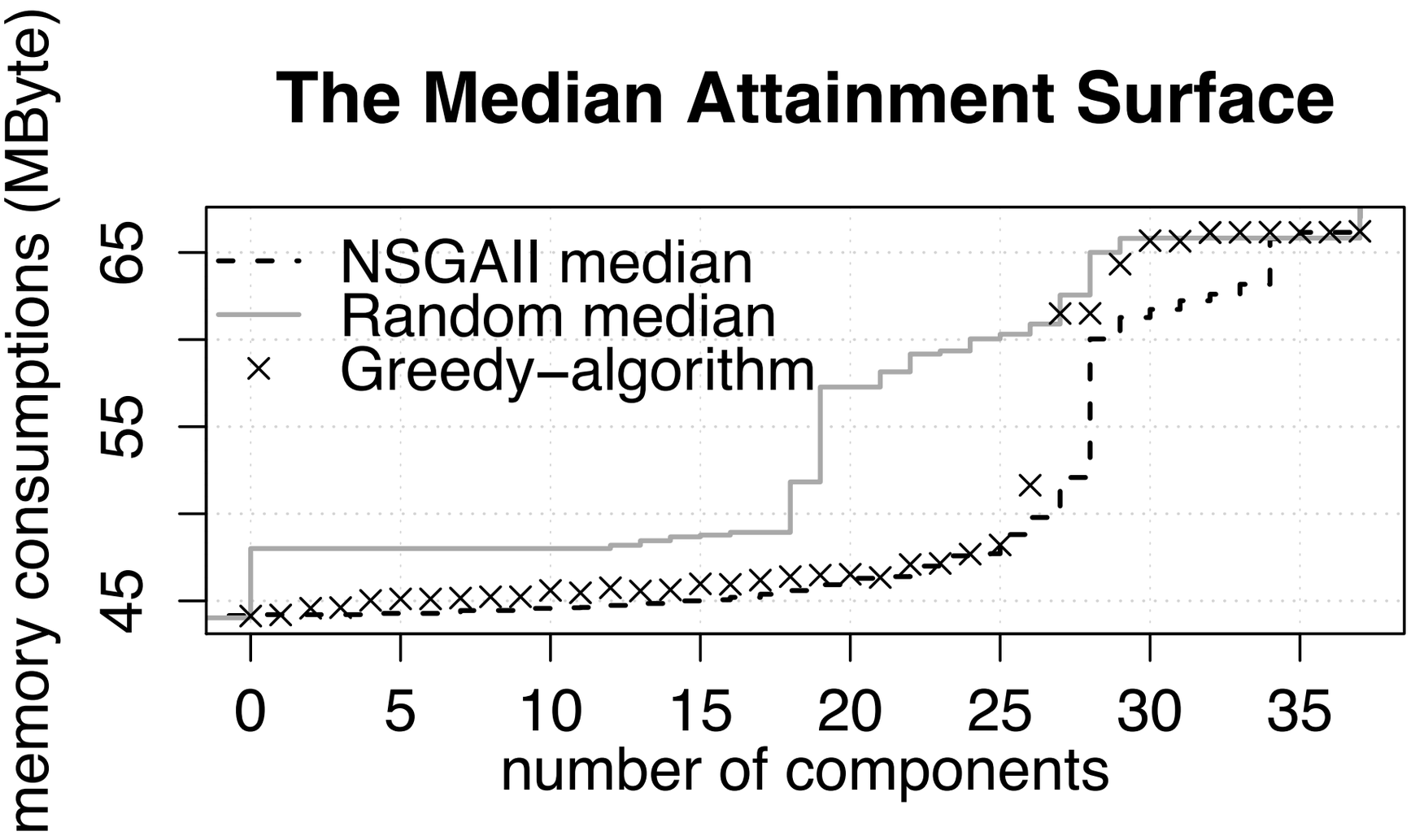
$$\Delta\vec{x}_{all} \approx 22.18\text{MB}$$

$$\text{Max}\Delta(\vec{x}_i) \approx 14.26\text{MB}$$

$$\text{Min}\Delta(\vec{x}_i) \approx 0.00\text{MB}$$

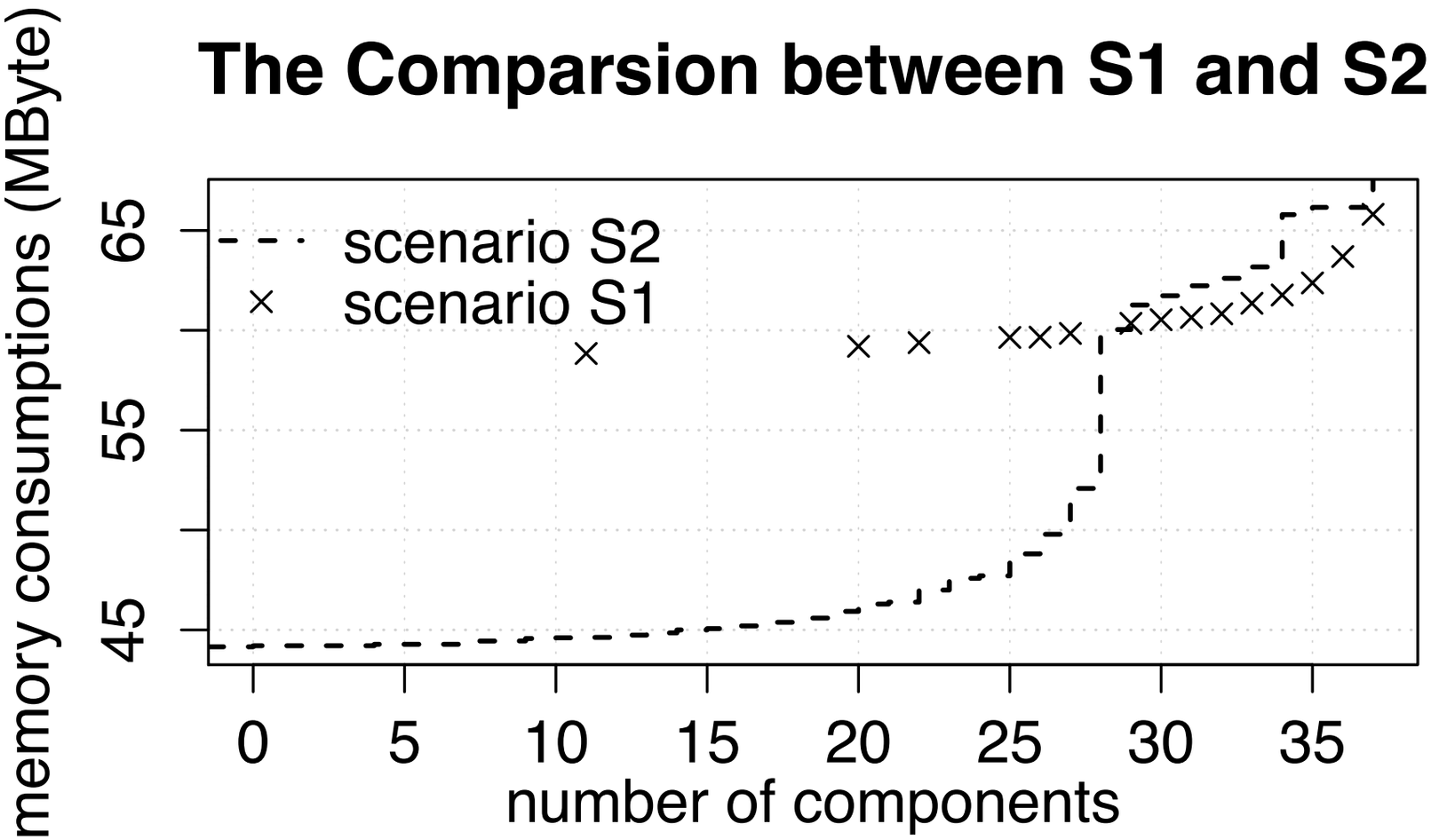
Answer to RQ2

RQ2.



Answer to RQ3

RQ3.



Introduction

SBSelector: Component Selection

Research Questions

Experiments and Results

Conclusions

Conclusions

We demonstrated that component selection problem can be treated as an instance of SBSE, and addressed it using search based techniques.

The results illustrate the trade-off between two types of user experiences.

The results also highlight some solutions that, when embedding the same number of components, our approach can reduce considerable memory consumption.

In one specific use case, SBSelector can find a solution that provides 16 more components while only increase negligible memory consumption.

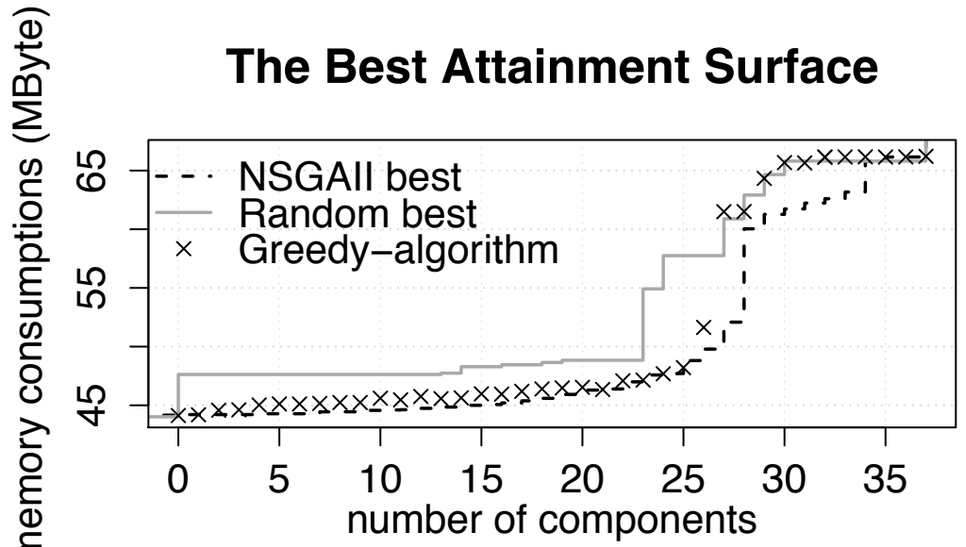
Kate | Get an Edge in Editing

$$\sum_i \Delta\vec{x}_i \approx 45.78\text{MB}$$

$$\Delta\vec{x}_{all} \approx 22.18\text{MB}$$

$$\text{Max}\Delta(\vec{x}_i) \approx 14.26\text{MB}$$

$$\text{Min}\Delta(\vec{x}_i) \approx 0.00\text{MB}$$



Thank You!