

Next-Gen Programming Interfaces and Compilers

Seminar Kick-off

Alexis Engelke Michael Petter Josef Weidendorfer

Chair of Data Science and Engineering (I25)
School of Computation, Information, and Technology
Technical University of Munich

2022-10-18

- ▶ Kick-off meeting 2022-10-18
- ▶ Literature research + derive structure
- ▶ Submit draft structure at latest 2022-11-21
- ▶ Discuss structure with advisor
- ▶ Write full draft paper 2022-12-05
- ▶ Peer-review two other papers 2022-12-19
- ▶ Incorporate feedback from peers and advisor
- ▶ Final submission of paper/slides 2023-01-24
- ▶ Presentations 2023-01-26/27

- ▶ Literature and sources
 - ▶ Finding literature and citable sources/references
- ▶ Writing a (seminar) paper
 - ▶ Structure, style, citing
- ▶ Presentation techniques
 - ▶ Structure, slide design, presentation style

Good to use

- ▶ Books, book chapters
- ▶ Papers (conf./journal)
- ▶ Published articles
- ▶ Manuals
- ▶ Websites with identifiable author
(cite with URL+access date)

Try to avoid

- ▶ Secondary Literature
- ▶ Wikipedia
- ▶ Facebook, etc.
- ▶ Advertisements
- ▶ Lecture slides
- ▶ Source code

- ▶ Starting points: IEEExplore, ACM DL, Google Scholar, ...
 - ▶ Select appropriate keywords
 - ▶ Many papers/books accessible freely via the university library
- ▶ Other starting point: your advisor
- ▶ Graph algorithms
 - ▶ Publications of the same author(s)
 - ▶ Publications at the same venue
 - ▶ Cites ... (listed references)
 - ▶ Cited by ...

1. Read title still relevant?
2. Read abstract still relevant?
3. Skim introduction/contributions still relevant?
 - ▶ Introduction sets framing
4. Skim through text and **figures** still interesting?
5. Read interesting sections

- ▶ Keep your references in BIBTEX files
- ▶ Also exportable from Google Scholar, ACM, ...
 - ▶ Caution: might be wrong (esp. G.Sc.) or contain irrelevant data

```
@inproceedings{lattner2004llvm,  
  title={{LLVM}: A compilation framework for  
    lifelong program analysis \& transformation},  
  author={Lattner, Chris and Adve, Vikram},  
  booktitle={Proceedings of the International  
    Symposium on Code Generation and Optimization},  
  series={CGO '04}  
  pages={75--86},  
  year={2004},  
}
```

- ▶ Abstract: Brief summary of area, problem, approach, key result
- ▶ Introduction: introduce area, problem, approach, key results, contributions, outline
- ▶ Background: if needed, describe prerequisites
- ▶ Main part (approach, evaluation, discussion, etc.)
- ▶ (*In a paper: Related Work – might come before main part*)
- ▶ Summary & outlook

- ▶ Factual, precise, focused, clear, simple
- ▶ Get to the point!
- ▶ Stay on topic, no story telling, . . .
- ▶ But: don't omit necessary prerequisites
- ▶ Make it easy for *the reader*

- ▶ Avoid forward references
- ▶ Avoid *I*, prefer *we* (or passive voice)
- ▶ *We* only described the authors, not the reader

- ▶ (Sub-)Sections to structure text
 - ▶ Allows reader to skip unimportant parts
 - ▶ No two headings without text in between
- ▶ Figures/tables: self-explaining with caption
- ▶ All figures/tables must be referenced in text
 - ▶ Allows reader to put figure in context
- ▶ Caption goes below figures, but above tables


- ▶ Text won't be perfect on first attempt
- ▶ What can be misunderstood?
- ▶ Cut out unnecessary words

- ▶ Fix grammar, spelling, punctuation, typography
 - ▶ Difference between -/—; hyphenation, quotes, ...
- ▶ Keep format standard and consistent
 - ▶ Fonts, colors, emphasis, ...
- ▶ Use *italics* (`\emph`), rarely **bold**, never underline

Three L^AT_EX mistakes that people should stop making?

1. Worrying too much about formatting and not enough about content.
2. Worrying too much about formatting and not enough about content.
3. Worrying too much about formatting and not enough about content.

– Leslie Lamport, 2000¹

¹LL Lamport. "How L^AT_EX changed the face of Mathematics". In: *DMV-Mitteilungen* 1 (2000), pp. 49–51. 

- ▶ All work that is not yours **must** be cited
 - ▶ Clearly describe source
 - ▶ But: no wrong/inaccurate attributions
- ▶ Citing styles:
 - ▶ Literal (direct) quote
 - ▶ indirect quote (rephrase) ←strongly preferred
- ▶ Exception: foundations can be assumed (generally first few Bachelor semesters)

The x86 architecture defines the register CR2 [1].

The x86 architecture defines the register CR2`\cite{intel2019man}`.

The x86 architecture defines the register CR2. It can be used with the instruction MOV. [1]

The x86 architecture defines the register CR2. It can be used with the instruction MOV.`\cite{intel2019man}`
(Absatz)

Valgrind [1] is a tool for run-time instrumentation.

Valgrind`\cite{nethercote2007}` is a tool for run-time instrumentation.

Other approaches [1,2,3] ...

Other approaches`\cite{foo,bar,baz}`
`\dots`

Presentation for the **audience!**

- ▶ What do you want the audience to take away?
(Not: what can I talk about!)
- ▶ What are the key points?
- ▶ How much content fits into the time slot?

- ▶ Motivation
 - ▶ Why is the topic relevant?
 - ▶ Background
 - ▶ Consider referencing information from previous talks
 - ▶ Concept
 - ▶ Evaluation
 - ▶ How good is the described concept?
 - ▶ Conclusions and outlook
-
- ▶ Important: avoid forward references
 - ▶ Restrict to important details
 - ▶ Use good/helpful examples

- ▶ Slides (Beamer)
 - ▶ For use during the talk
 - ▶ Good to prepare
 - ▶ *Backup slides* as preparation for questions
- ▶ Whiteboard, blackboard
 - ▶ Permanently needed information
 - ▶ Answering questions
- ▶ Hardware, demonstrators, etc.

- ▶ Check possibilities in advance

- ▶ Prepare slides, etc.
- ▶ Do a dry-run
 - ▶ Always recommended
 - ▶ Helps with uncertainty and time estimation
- ▶ Prepare on-site
 - ▶ Laptop, Beamer, laser pointer, clock, etc.

- ▶ Speak freely
- ▶ Don't go too fast/slow
- ▶ Stay in contact with the audience
 - ▶ Eye contact, position, etc.
- ▶ Usually at least 1 minute per slide
- ▶ Stay in time limit
 - ▶ Optional slides can fill time
 - ▶ Regularly consult a watch

- ▶ **Stay calm**

- ▶ One topic per slide
- ▶ Avoid text
 - ▶ ≤ 8 lines
- ▶ Prefer graphics/illustrations
- ▶ No unused points
 - ▶ Cover everything on the slides in your talk

- ▶ Title page
 - ▶ Title, name, institution, date, location
- ▶ On every other slide: number and title
- ▶ Conclusion
 - ▶ All important points on one slide

- ▶ **Black on white**
- ▶ **Black on white**
- ▶ Sufficient contrast
- ▶ Use colors sparingly, but systematically
- ▶ Be careful with gradients
- ▶ No annoying backgrounds (wave textures, etc.)
- ▶ **Anomations only with sufficiently added value**

- ▶ Double-check text for typos, etc.
- ▶ Use a readable, sans-serif font
- ▶ Prefer vector graphics (or images with a high resolution)
- ▶ Avoid screenshots/scans
- ▶ Citations: if critical, use footnote
 - ▶ No end notes and [12]-style references

- ▶ Listings only with a sufficiently large value

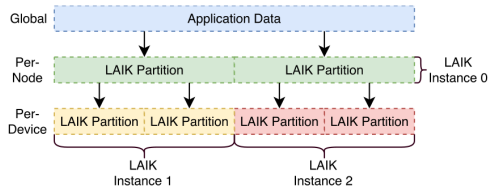
```
\begin{frame}
\frametitle{Die Anti-Folie}
\begin{figure} [ht]
  \centering
  \includegraphics[width=0.95\textwidth]{pictures/antifolie.jpg}
  \caption{Werbe-Folie. Foto von Flickr-Benutzer niallkennedy
    (https://www.flickr.com/photos/niallkennedy/58697220/sizes/l/)}
  \label{fig:gliederung}
\end{figure}
\end{frame}
```

Figure: Screenshot of code with insufficient resolution

LAIK (5) – Hierarchische Partitionierung

- multiple Partitionierung auf verschiedenen Ebenen
- Beispiel: inter/intra-node
- sinnvoll für Exascale, heterogene Systeme

- Veränderung des Indexraums muss möglich sein!



```
#include "laik-backend-mpi.h"
int main(int argc, char* argv[])
{
    Laik_Instance* inst = laik_init_mpi(&argc,&argv);
    Laik_Group* world = laik_world(inst);

    // allocate global 1d double (8 bytes) array: 1 mio entries
    Laik_Data* a = laik_alloc_1d(world, 8, 1000000);

    // initialize at master (others do nothing)
    laik_set_new_partitioning(a, LAIK_PT_Master, LAIK_AP_WriteOnly);
    double* base; uint64_t count;
    laik_map(a, LAIK_DL_CANONICAL, (void**) &base, &count);
    for(uint64_t i = 0; i < count; i++) base[i] = (double) i;
}
```

Figure: Example for showing source code

```
#include "laik-backend-mpi.h"
int main(int argc, char* argv[])
{
    Laik_Instance* inst = laik_init_mpi(&argc,&argv);
    Laik_Group* world = laik_world(inst);

    // allocate global 1d double array: 1 mio entries
    Laik_Data* a = laik_alloc_1d(world, 8, 1000000);
    // initialize at master (others do nothing)
    laik_set_new_partitioning(a, LAIK_PT_Master,
                              LAIK_AP_WriteOnly);
    double* base; uint64_t count;
    laik_map(a, LAIK_DL_CANONICAL, (void**)&base, &count);
    for(uint64_t i = 0; i < count; i++)
        base[i] = (double) i;
}
```

Figure: Example for showing source code

- ▶ Bring your point to the audience – written or spoken
- ▶ Good literature as starting point
- ▶ Logical structure for paper and presentation
- ▶ Make it easy for audience to get information
- ▶ Presentation: good preparation is important

- ▶ Chance to learn 😊