TDDD89

Lecture 3 - Literature search and evaluation

Christoph Kessler

LINKÖPINGS UNIVERSITET

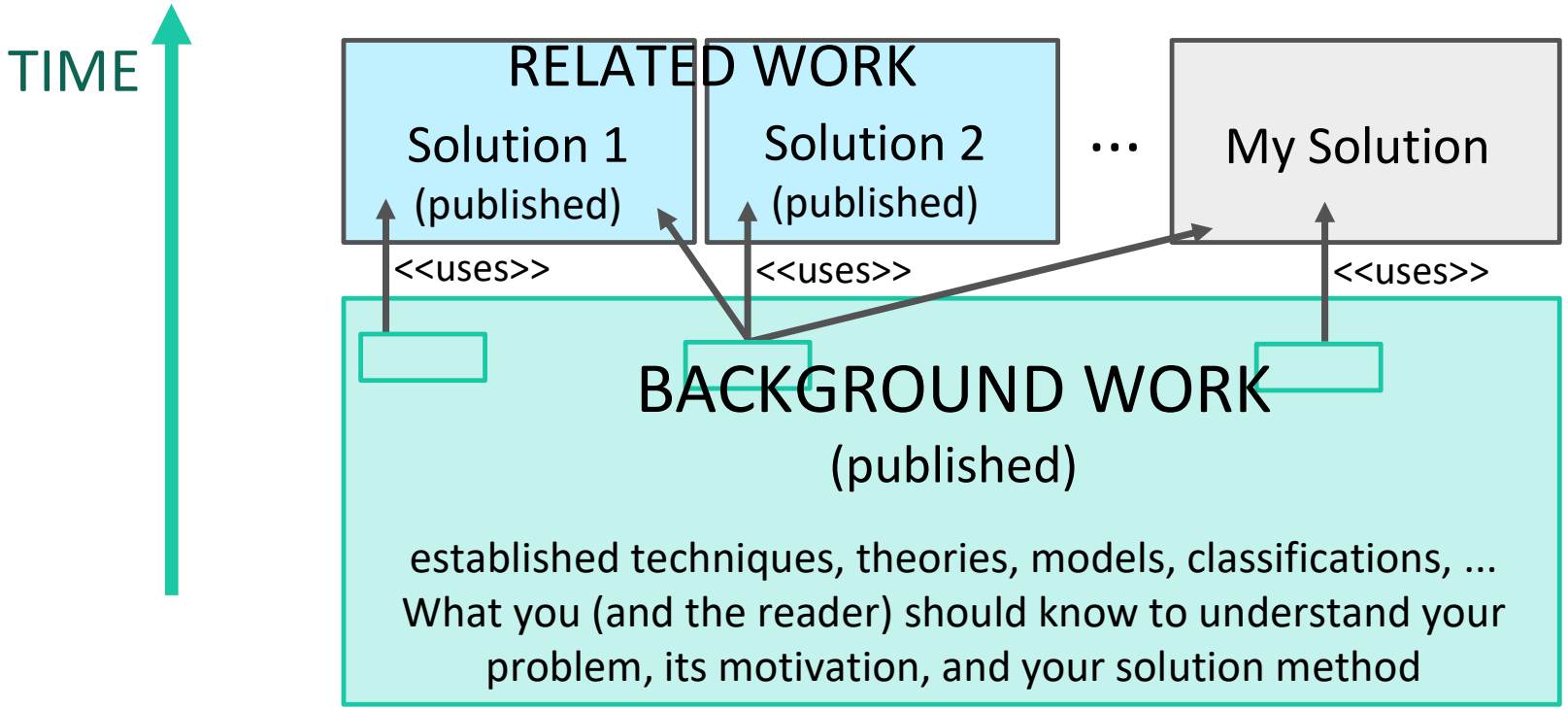
Re-SEARCH

Finding, assessing and referencing relevant literature

- Background literature
- Related work

Background vs. Related Work

- In a nutshell: (here, for a thesis of problem-solving type)



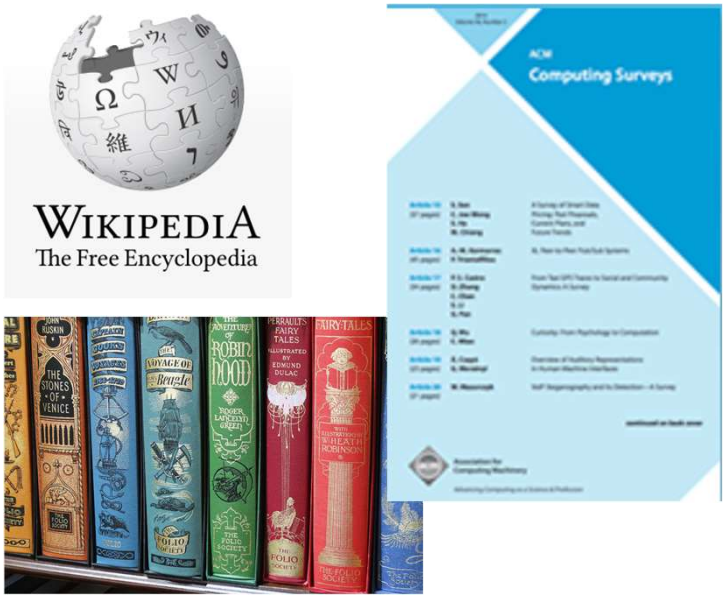
What you, in your thesis work, *use* or *build upon* is *background*, not related work.
Related work are others who solved the same/similar problem based on *same/similar* background.

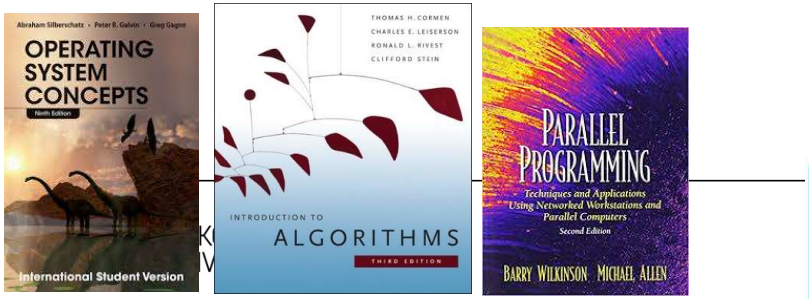
Background Literature vs. Related Work?

- **Related work**: Studies that are *similar* in style and objective to what you are doing in your own thesis work, i.e., work that you can *compare to*.
  - Published work addressing the same or a slightly different problem, e.g., for a different processor architecture, programming language, or for a variant of the algorithmic problem considered.
  - Read, digest, describe, **compare**. ~ 1 paragraph per paper.
  - Relate your own experimental evaluations and conclusions to those works and explain the similarities and differences, strengths and weaknesses, thereby leading to a proper related work discussion in your thesis and possibly stronger conclusions.
- Manuals, survey articles, books etc. for theory, systems and techniques that you ***use*** for your work are **background literature**, *not* related work (but should nevertheless be cited).

How to find related work and relevant background literature?

Finding information

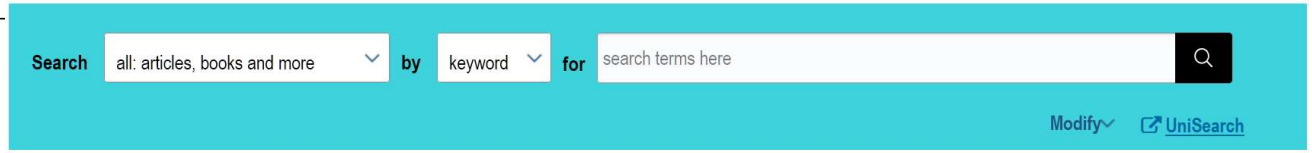




Learn about the subject area:

- Start from e.g. Wikipedia (with care), books, survey articles and previous course material
- Extract *keywords* that you can use when searching papers.
- Use Google Scholar and Unisearch first, specific publications second

Search articles and books https://liu.se/en/library



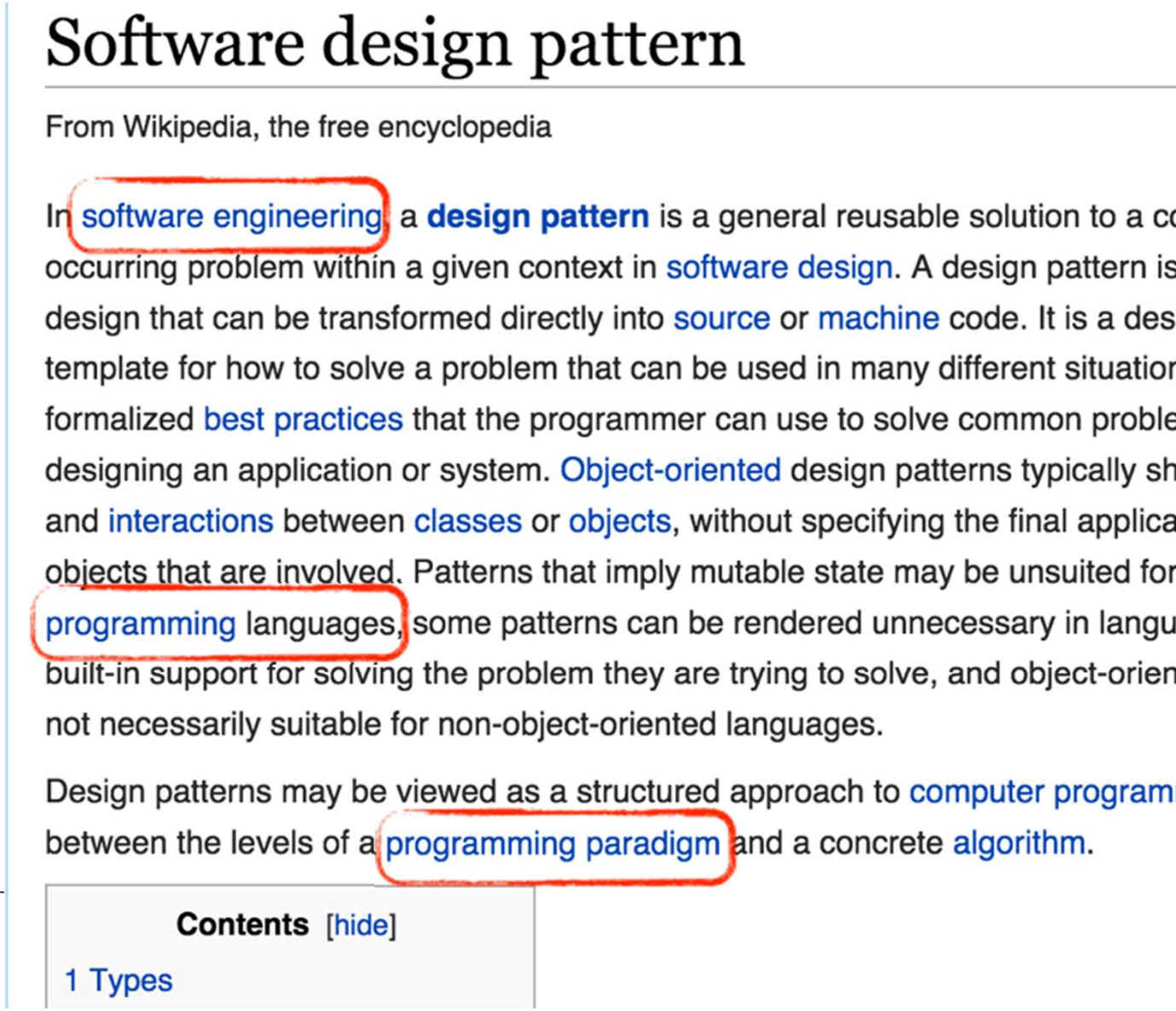
Wikipedia??

- Not stable
- Anonymous authors
- May be biased
- May be incomplete
- Varying quality
- Unclear quality control

→ **NOT CITEABLE!**

But still useful for getting an *early* overview of a new topic area

- by following (and reading) given ***references***
- finding appropriate **keywords** and their **synonyms** for better search



Software design pattern

From Wikipedia, the free encyclopedia

In software engineering, a **design pattern** is a general reusable solution to a co occurring problem within a given context in software design. A design pattern is design that can be transformed directly into source or machine code. It is a des template for how to solve a problem that can be used in many different situation formalized best practices that the programmer can use to solve common proble designing an application or system. Object-oriented design patterns typically sh and interactions between classes or objects, without specifying the final applica objects that are involved. Patterns that imply mutable state may be unsuited for programming languages, some patterns can be rendered unnecessary in langu built-in support for solving the problem they are trying to solve, and object-orien not necessarily suitable for non-object-oriented languages.

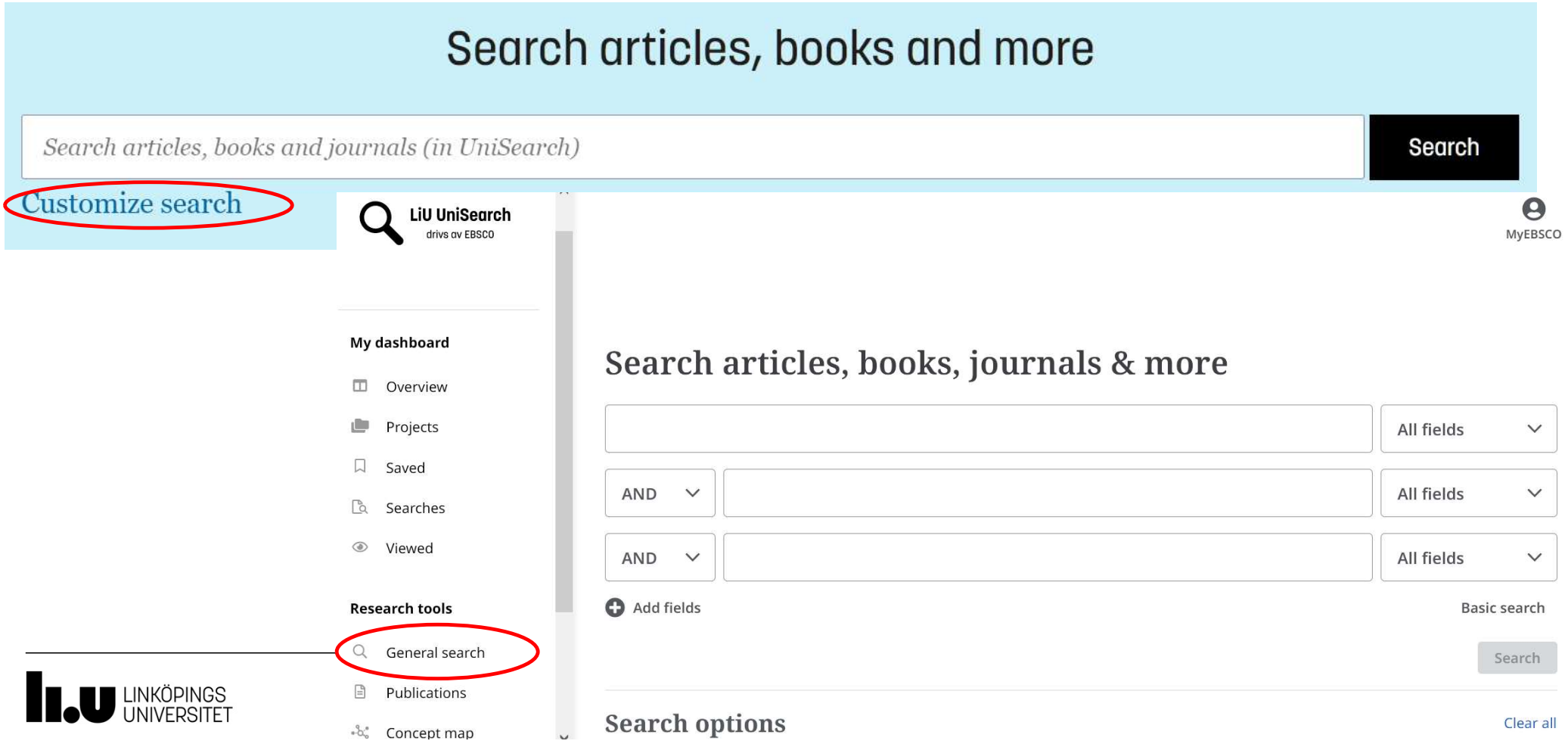
Design patterns may be viewed as a structured approach to computer program between the levels of a programming paradigm and a concrete algorithm.

Contents [hide]

1 Types

Stå på giganters axlar

Unisearch https://liu.se/en/library

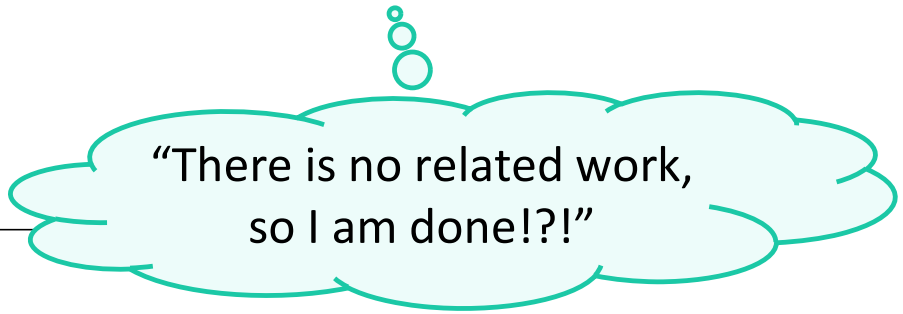


LINKÖPINGS UNIVERSITET

How to find related work

”HLA active probing runtime performance requirements in a Wide Area Network”

{ }



Smart Searching

HLA	simulation
active probing	fault detection and localization
runtime performance requirements	latency, throughput
in a Wide Area Network	IP networks

→ Find and try **synonyms**;
generalize over (company/domain-local) jargon.

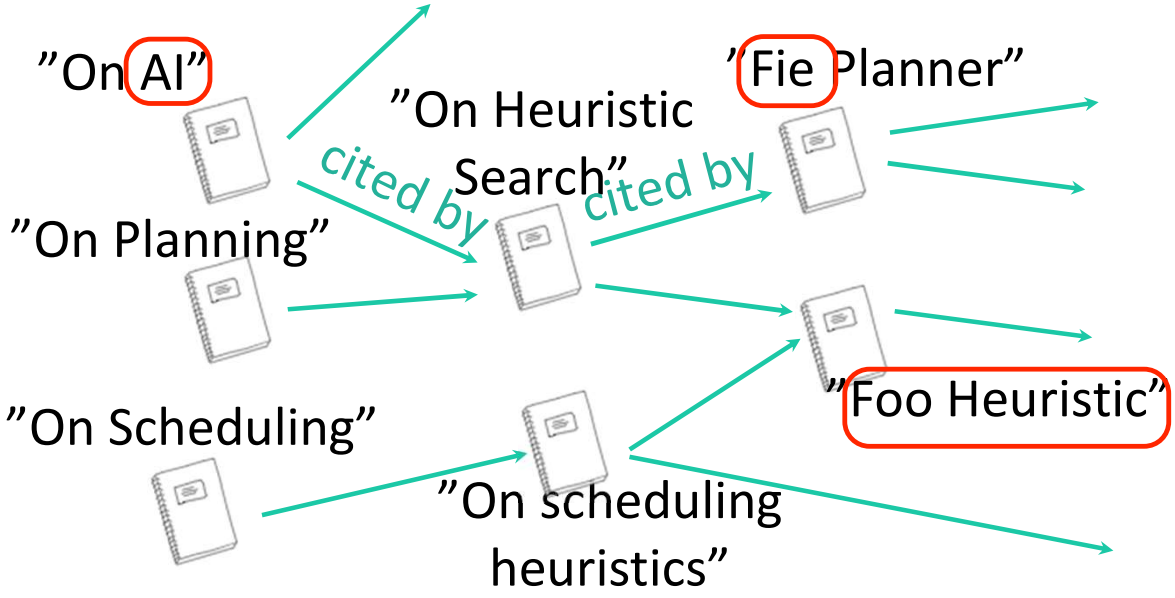
Engineering information vs Scientific information

	Engineering	Science
Questions	How to solve a problem	How to explain something
Reliability	Working solutions, proven theories	Peer-reviewed publications, Cited work
Sources	White papers, software projects, peer-reviewed publications, patents	Peer-reviewed publications

Iterative search

Keywords, synonyms

Search entry points



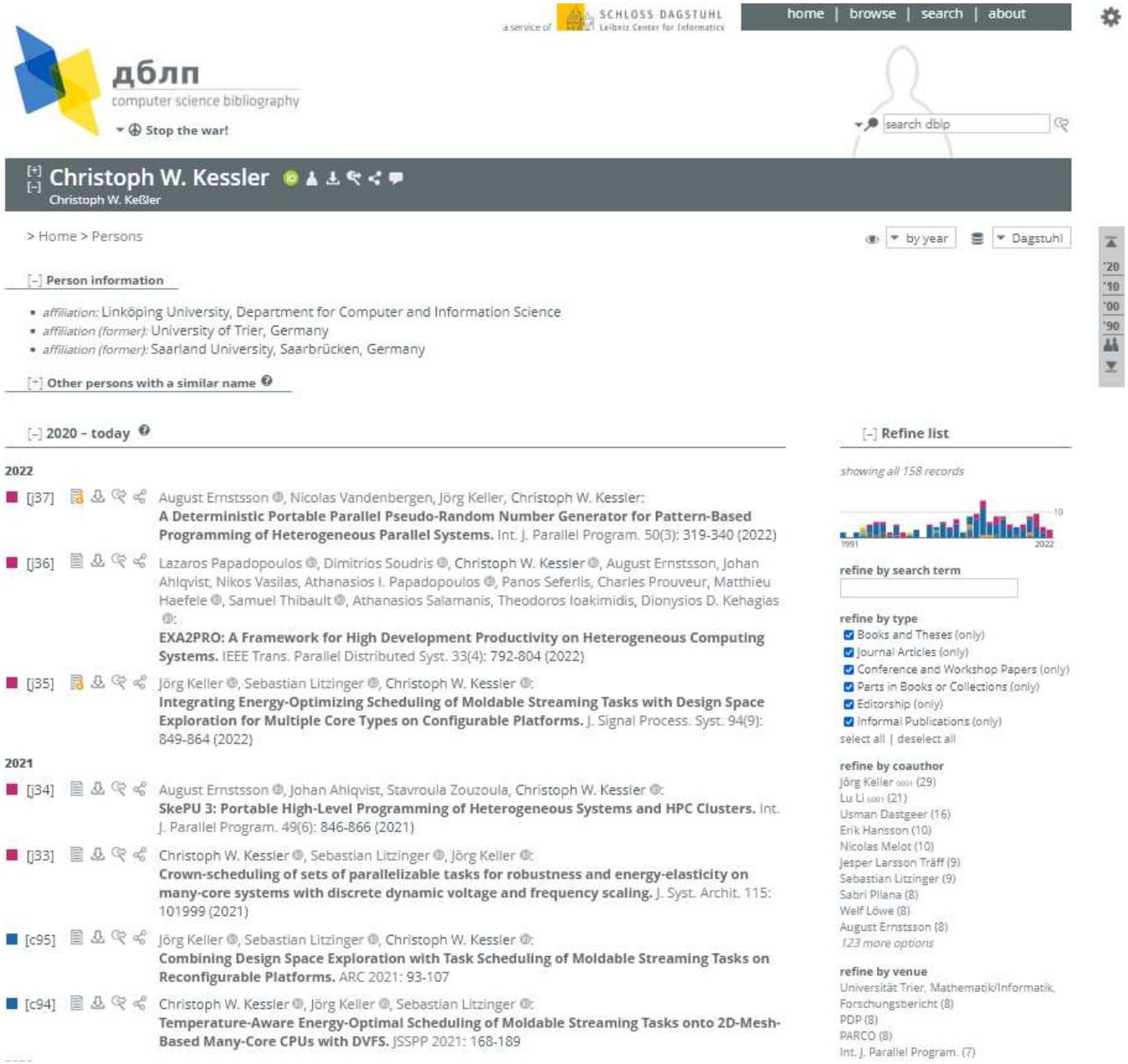
Recursive search through the world-wide digraph of publications – following the “cited-by” relation in both directions

1 2 3

Search by Author

"This paper by author *X* was right on target. Did *X* also write other papers about this or related topics?"

- Author homepage (if existing)
- Google Scholar
- DBLP: https://dblp.org



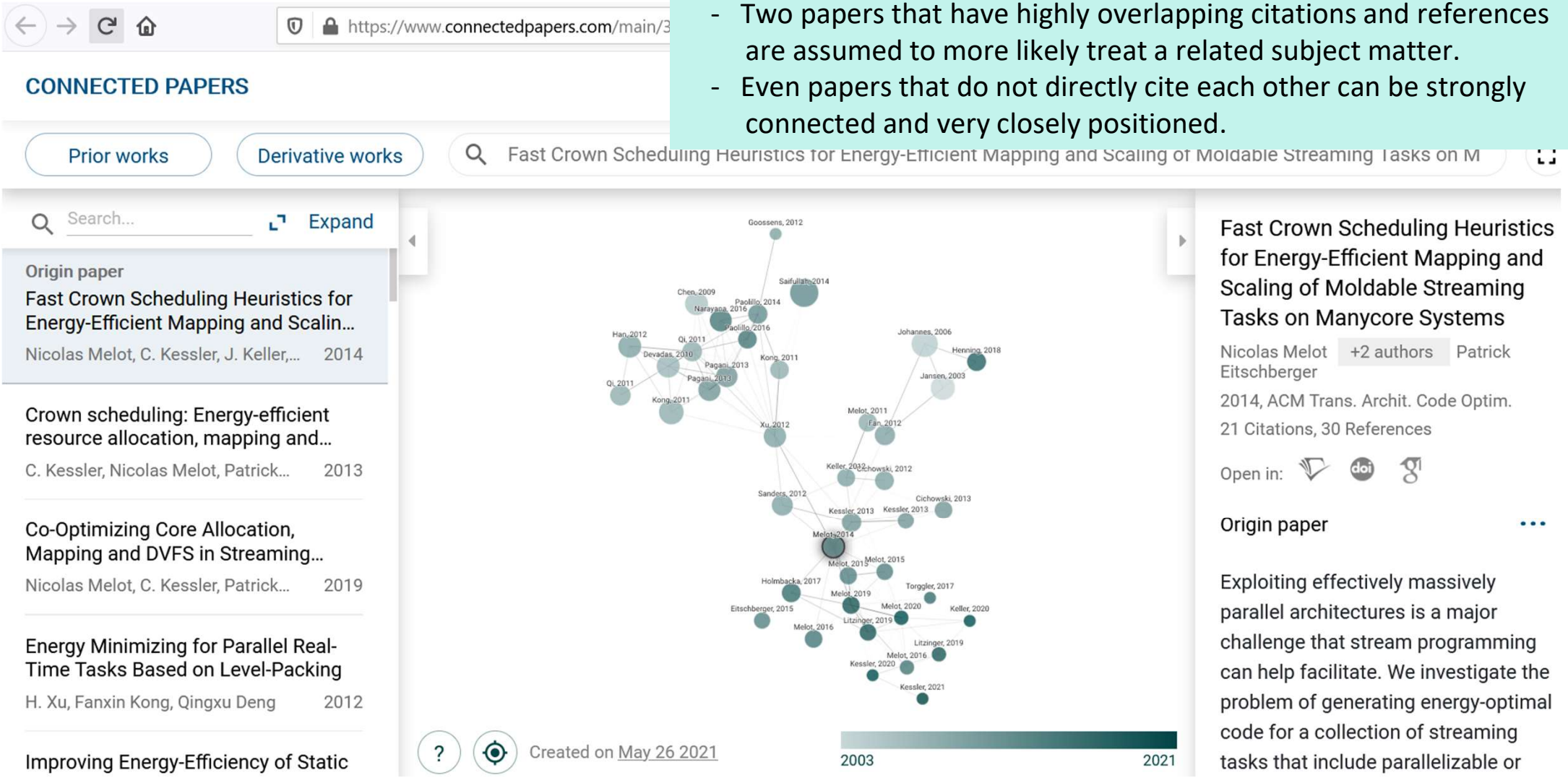
LINKÖPINGS UNIVERSITET

Search tool: connectedpapers.com

Input: A paper (DOI, title or title keywords)

Found papers are arranged according to their “similarity” based on ***co-citation*** **and** ***bibliographic coupling***

- Two papers that have highly overlapping citations and references are assumed to more likely treat a related subject matter.
- Even papers that do not directly cite each other can be strongly connected and very closely positioned.



Scientific publishing

Primary studies

Secondary studies

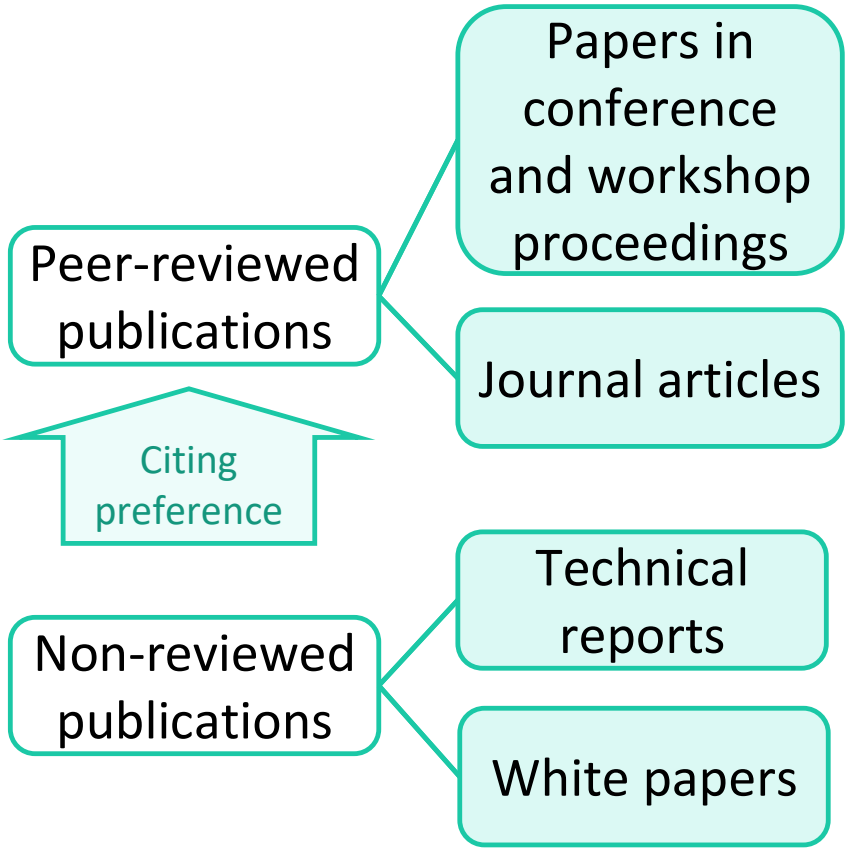
Text books

"What"

Citing preference towards primary studies

Scientific publishing

”How”



Publication types

can often be inferred already from a descriptive title alone –
without even accessing the paper!

P. Kruchten, H. Obbink, and J. Stafford. The past, present, and future for software architecture. *IEEE Software*, 23(2):22–30, March–April 2006.

***Position paper or survey paper**, i.e., secondary study, no new research results. Shares experience on software architecture research and development. Published in a journal.*

Publication types

T. K. Paul and M. F. Lau. A systematic literature review on modified condition and decision coverage. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, SAC '14, pages 1301–1308, New York, NY, USA, 2014. ACM.

***Systematic Literature Review**, secondary study.*
Published at a conference.

Publication types

C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering*. Springer Berlin Heidelberg, 2012.

*Guidelines **textbook** on empirical methods in Software Engineering.*

OK as reference to background literature, but primary / secondary studies are to be preferred.
Textbooks are never related work.

Publication types

I. Maier, T. Rompf, and M. Odersky. Deprecating the observer pattern. Technical report, École Polytechnique Fédérale de Lausanne, 2010.

***Technical report**, non-reviewed publication.*
No empirical support for claims, but suggestions of an architecture.

Older technical reports can be hard to get hold of.
Find a published (conference, journal, book chapter) version of this work that can be cited instead.

Publication types

A. Nilsson, J. Bosch, and C. Berger. Visualizing testing activities to support continuous integration: A multiple case study. In G. Cantone and M. Marchesi, editors, *Agile Processes in Software Engineering and Extreme Programming*, volume 179 of *Lecture Notes in Business Information Processing*, pages 171–186. Springer International Publishing, 2014.

***Case study**,*
peer-reviewed publication in a conference proceedings volume
(in a Springer series of conference volumes and other edited books)

Publication types

J. Andrews, L. Briand, and Y. Labiche. Is mutation an appropriate tool for testing experiments? In *Proceedings of the 27th International Conference on Software Engineering*, ICSE 2005, pages 402–411, May 2005. IEEE Computer Society.

***Experimental study**,*
peer-reviewed publication presented at a conference and published in proceedings from the conference.

Evaluating Quality of Literature

(a) Quality of the technical/scientific contributions
(b) Quality of the publication channel (trust-based)

What are the paper's results?

Result Type	How Validated?	Quality Criteria
Procedure/technique	Formal proofs, experiments, statistical support	Inclusion of (sound) proofs, selection of benchmark programs, proper use of statistics
Descriptive models		Properly accounting for reality
Experience reports	Interviews, observations, usage data	Real systems and people

Many papers contain an explicit list of **contributions** at the end of the introduction section, which summarize the main results.

In most domains of CS/CE, this usually replaces the original explicit research questions (= internal working material). Still listing explicit RQs is most typical for papers in Software Engineering and for *theses*.

What are *strong* results?

Real systems and **_proper_ analysis**

Table 6. Types of research validation represented in ICSE 2002 submissions and acceptances			
Type of validation	**Submitted**	**Accepted**	**Ratio Acc/Sub**
Analysis	48 (16%)	11 (26%)	23%
Evaluation	21 (7%)	1 (2%)	5%
Experience	34 (11%)	8 (19%)	24%
Example	82 (27%)	16 (37%)	20%
Some example, can't tell whether it's toy or actual use	6 (2%)	1 (2%)	17%
Persuasion	25 (8%)	0 (0.0%)	0%
No mention of validation in abstract	84 (28%)	6 (14%)	7%
TOTAL	300(100.0%)	43 (100.0%)	14%

M. Shaw: "Writing good software engineering research papers: Minitutorial." In *Proceedings of the 25th International Conference on Software Engineering*, ICSE '03, pages 726–736, Washington, DC, USA, 2003. IEEE Computer Society.

Contributions, explicitly stated (by the authors)

old violations on average and achieves 19.09% and 28.53% total energy savings in comparison with two state-of-the-art techniques.

The contributions of this paper are listed as follows.

1) An accurate and efficient temperature model is developed, which achieves an average of over 99% temperature prediction accuracy with greatly reduced computation time, compared to a widely recognized temperature simulator.
2) A systematical approach for energy-efficient application mapping and scheduling on MPSoC platforms is proposed, in which chip temperature profile and lifetime reliability issue are explicitly taken into consideration simultaneously and are well controlled. Performance evaluations on extensive synthetic and real-world applications prove the effectiveness of the proposed approach.

The rest of this paper is organized as follows. Section II discusses related work; Section III presents the system models and reliability modeling techniques; Section IV uses an

What is new here?
What are the main results?
Are the results good?
Are the results relevant?

— Source: W. Liu *et al.*: Energy-efficient application mapping and scheduling for lifetime guaranteed MPSoCs. *IEEE Trans. Computer-aided Design of Integr. Circuits and Syst.*, vol. 38 no. 1, Jan. 2019

How to evaluate papers

- Relevance = *f* (title, year, abstract/contributions, type, venue, #citations)
  - The more specific the paper, the fewer citations?
- Literature reviews: meta studies
- Publication types: journals, conferences, book chapters
- Trust is inherited from the journal's, conference's or publisher's **reputation**
  - ISI/Web-of-Science or Scopus listed journals, Norwegian level ranking, … see liu.se/en/library/publishing
  - CORE ranking of conferences and journals www.core.edu.au
  - Beware of predatory publishers, fake conferences and fake journals!
    – check *Beall's list* (albeit somewhat outdated) if unsure → beallslist.net
    - Do not cite work published in such questionable venues!!!
- Read with critical eyes.
  - Refer to the real main results of the paper (describe in your own words), not copy what the authors have themselves written in the abstract or introduction

Predatory Publishing: What Are the Alternatives to Beall's List? Commentary, *The American Journal of Medicine* 131(4), Elsevier 2018. https://doi.org/10.1016/j.amjmed.2017.10.054

Example: ISI ranking of journals

http://www.robertfeldt.net/advice/se_venues/

Latest update: 2016-10-26 11:23.41

ISI SE JOURNALS (RANKED)

Journal	YC	IQR	MC	P0
1. Transactions on Software Engineering	**6.7**	3.4-13.0	10.7	1.2
2. Empirical Software Engineering	**5.9**	2.6-11.0	8.3	1.3
3. Transactions on Software Engineering and Methodology	**5.0**	2.3-10.5	8.1	0.0
4. Automated Software Engineering	**4.7**	2.0-8.7	6.6	7.2
5. Information & Software Technology	**4.5**	2.5-8.5	7.0	1.0
6. Requirements Engineering	**4.2**	2.5-8.6	6.4	1.1
7. Software & Systems Modeling	**4.0**	2.0-7.3	6.0	3.1
8. Software Quality Journal	**3.7**	1.5-6.8	5.0	2.6
9. Journal of Systems and Software	**3.5**	1.7-6.6	5.2	1.9
10. Journal of Software: Evolution and Process	**2.3**	1.0-4.5	4.2	8.3
11. Software Testing, Verification & Reliability	**2.2**	1.0-5.5	6.3	5.3
12. Software: Practice & Experience	**2.0**	0.7-4.2	4.9	6.3
13. IET Software	**1.5**	0.7-2.8	2.2	5.4
14. Int. Journal of Software Engineering and Knowledge Engineering	**0.8**	0.2-1.5	1.1	23.4

What about white papers / other stuff?

- Not peer-reviewed
- Use to support existence: ”There are several implementations of Flux controllers”
- Not to support claims and propositions: ”Flux controllers are more user friendly than Flax controllers”
- Company white-papers, data sheets, manuals etc.
  - URL will likely change within a few years, or the paper be removed/replaced

- Gray-zone: *ArXiv* for preprints – may not be peer-reviewed (exceptions exist)
  - has a stable URL, but everyone can upload a paper to *ArXiv* – thus far less trustworthy than published material
  - **If published, cite the journal / conference version instead**.

Citing the right reference (1)

”*Software product lines* are related software products that are customized to different customers [1].”

[1] Kästner, C., Apel, S., and Kuhlemann, M. Granularity in software product lines. In *Proceedings of the 30th International Conference on Software Engineering*, ICSE ’08, pages 311–320, New York, USA, 2008.

Not the main result of [1]

[2] Pohl, K., Böckle, G., and van der Linden, F. J. (2005). *Software product line engineering: foundations, principles and techniques*. Springer Science & Business Media.

Cite the book by Pohl *et al*. [2] instead

Citing the right reference (2)

”As an example algorithm, let us consider *Quicksort* [1].”

[1] Cormen, T., Leiserson, C., Rivest, R., Stein, C.: *Introduction to Algorithms*, 3rd edition. MIT Press, 2009.

A textbook

[1] Hoare, C. A. R. (1961). "Algorithm 64: Quicksort". *Comm. ACM*. 4 (7): 321.

The original publication

Referencing with care:
Build and preserve trust in your work

By referencing a paper, one implicitly adds *trust* / *credit* to that work.

- High citation counts are widely considered a scientific merit of the author(s) and the venue, and used as a proxy metric for scientific quality
  - Widely used in academia e.g. when ranking applicants for academic positions, for seeking internal promotion, and for scientists' salary reviews
  - Used by journals in advertising (”impact factor”)
  - Usage as a real-quality indicator is highly debatable

- Citing a good paper / conference / journal / publisher with good reputation will increase others' trust in your own work.
- Citing a bad paper / conference / journal / publisher (e.g. a predatory publisher) will add some ”smell” to your work.

Referencing with care:
Play the strongest cards you can find!

There is an unofficial but well-known relative-quality ranking of scientific publication forms/channels, e.g.:

- *journal article* [1] trumps *conference paper* (but exceptions exist, esp. in CS and CE!), yet with big differences within each category, see e.g.:
  - Journal rankings, e.g. at CORE
  - CORE conference rating: https://www.core.edu.au/conference-portal
    A* (strongest; hardest competition), A, B, C, no classification (weakest)

CORE2023 Summary:
A* - 7.46% of 804 ranked venues
A - 14.55% of 804 ranked venues
B - 27.74% of 804 ranked venues
Australasian B - 0.75% of 804 ranked venues
C - 47.14% of 804 ranked venues
Australasian C - 2.36% of 804 ranked venues
Other - 164 total

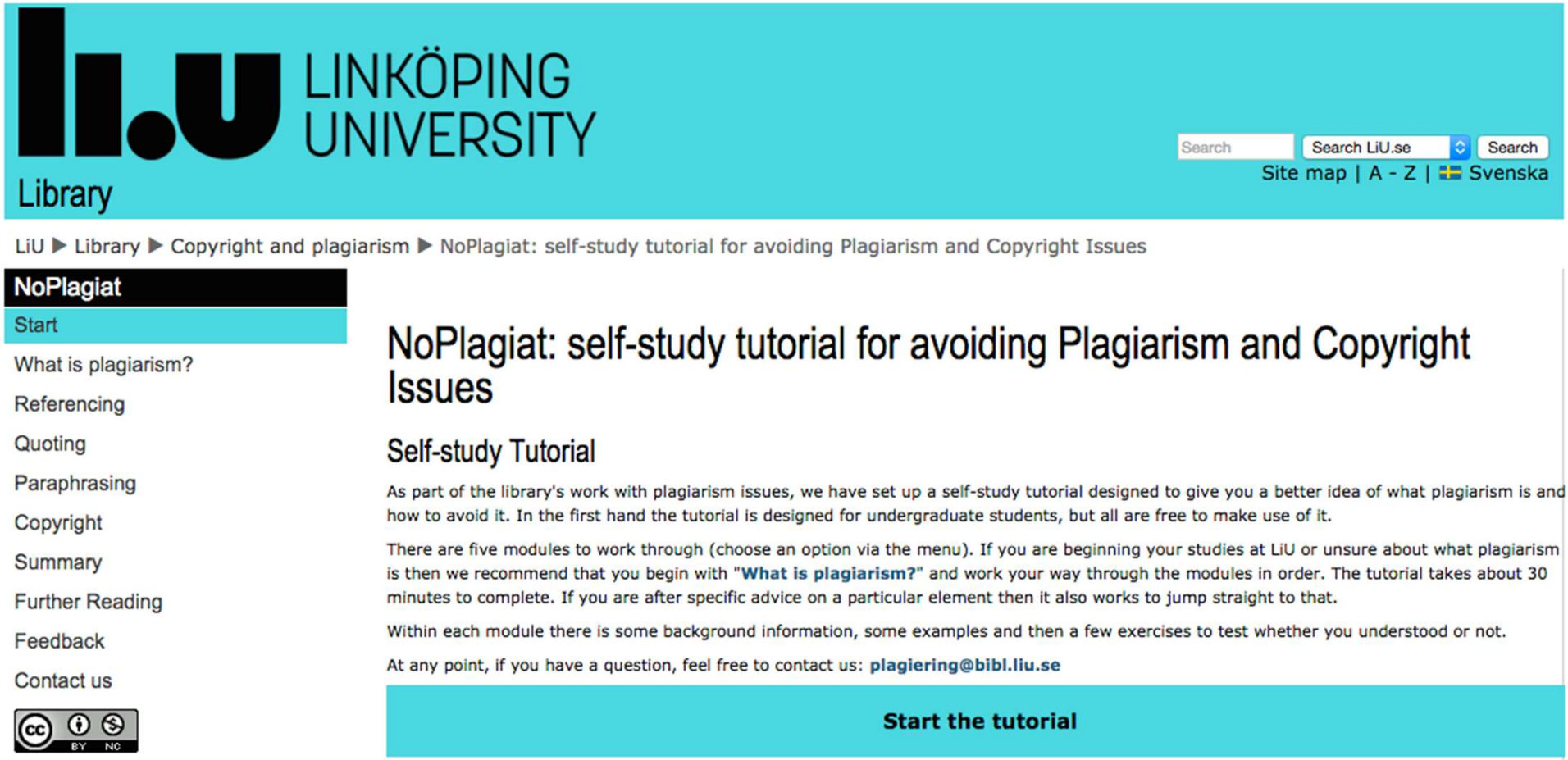
- *conference paper* trumps *workshop paper (formally published)*
- *journal/conference/workshop paper* trumps *thesis* or *technical report*
- *original paper/article* trumps *textbook*
- *book / book chapter* trumps *thesis / technical report*
- all these trump *potentially unreviewed* or *unpublished* work (ArXiv [1], www, blogs, Wikipedia) and work in *predatory venues* (smell)

---

[1] Extended versions of conference papers (≥30% new contents) sometimes appear later in some journal.

ArXiv: some workshops with peer-review process do publish in *ArXiv* or similar open-access archives only
→ giving the workshop name, if known, can add weight to an *ArXiv* reference

Plagiarism and copyright



Available as a quiz on the Lisam page for our course

Using image without reference

Plagiarism + copyright violation

Using image with reference

Copyright violation

Using own image / CC image with reference

OK!

From *LiU News*, article by Gunilla Pravitz 2011-03-29, https://old.liu.se/liu-nytt/arkiv/nyhetsarkiv/1.262645?l=en

LiU News

Damages claimed for pictures in degree project

A LiU alumnus risks having to pay SEK 4,000 for copyright infringement concerning pictures used in a degree project he did eleven years ago.

The work dealt with how to identify various deciduous trees in winter, and the text was illustrated with 12 photos and 12 drawings from an existing book on the subject. The project was approved in 2000.

> "The teaching student provided the book's title under the illustrations, but not the author's name nor had he asked the author for permission to use the pictures," says Martin Putsén, a lawyer with the University administration.

It could now cost the former student SEK 4,000. The author has claimed copyright infringement damages through **SLFF, the Swedish Association of Educational Writers**, after discovering the unlicensed use. SLFF has also demanded that the degree project no longer be made available in the University library.

Using citations and references properly

Citation: annotation / symbol / cross-reference in the running text, e.g. [23]

- Indicates where the information or material came from.
- Unique in the thesis/article,
  but intentionally kept very short to not disrupt the flow of reading.
- Usually indirect (refers to an entry in the References list).

Reference: complete bibliographical information for each referenced work in the list of references at the end of the thesis/article.

References

There are a number of conventions of how to use references properly:
use in-text references or outside-text references consistently.
IEEE has a good standard for this.

~~[1] has studied software design patterns~~

Odersky et al. [1] have studied software design patterns.
Odersky et al. (2010) have studied software design patterns.
Odersky et al. have studied software design patterns [1].

Paraphrasing

= expressing someone else's text **with your own words**.

Do not copy verbatim from published papers!

Over a quarter of the ICSE 2002 abstracts give no indication of how the paper's results are validated, if at all [1].

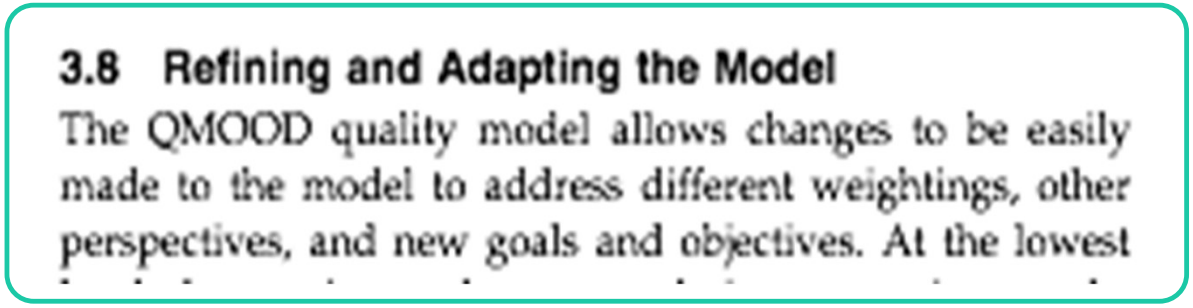
4.2 Which of these are most common?

Alas, well over a quarter of the ICSE 2002 abstracts give no indication of how the paper's results are validated, if at all. Even when the abstract mentions that the result

[1] M. Shaw. Writing good software engineering research papers: Minitutorial. In *Proceedings of the 25th International Conference on Software Engineering*, ICSE '03, pages 726–736, Washington, DC, USA, 2003. IEEE Computer Society.

Quotations and Citations

Bansiya and Davis claim that the QMOOD model may address ”different weightings, other perspectives, and new goals and objectives” [1].



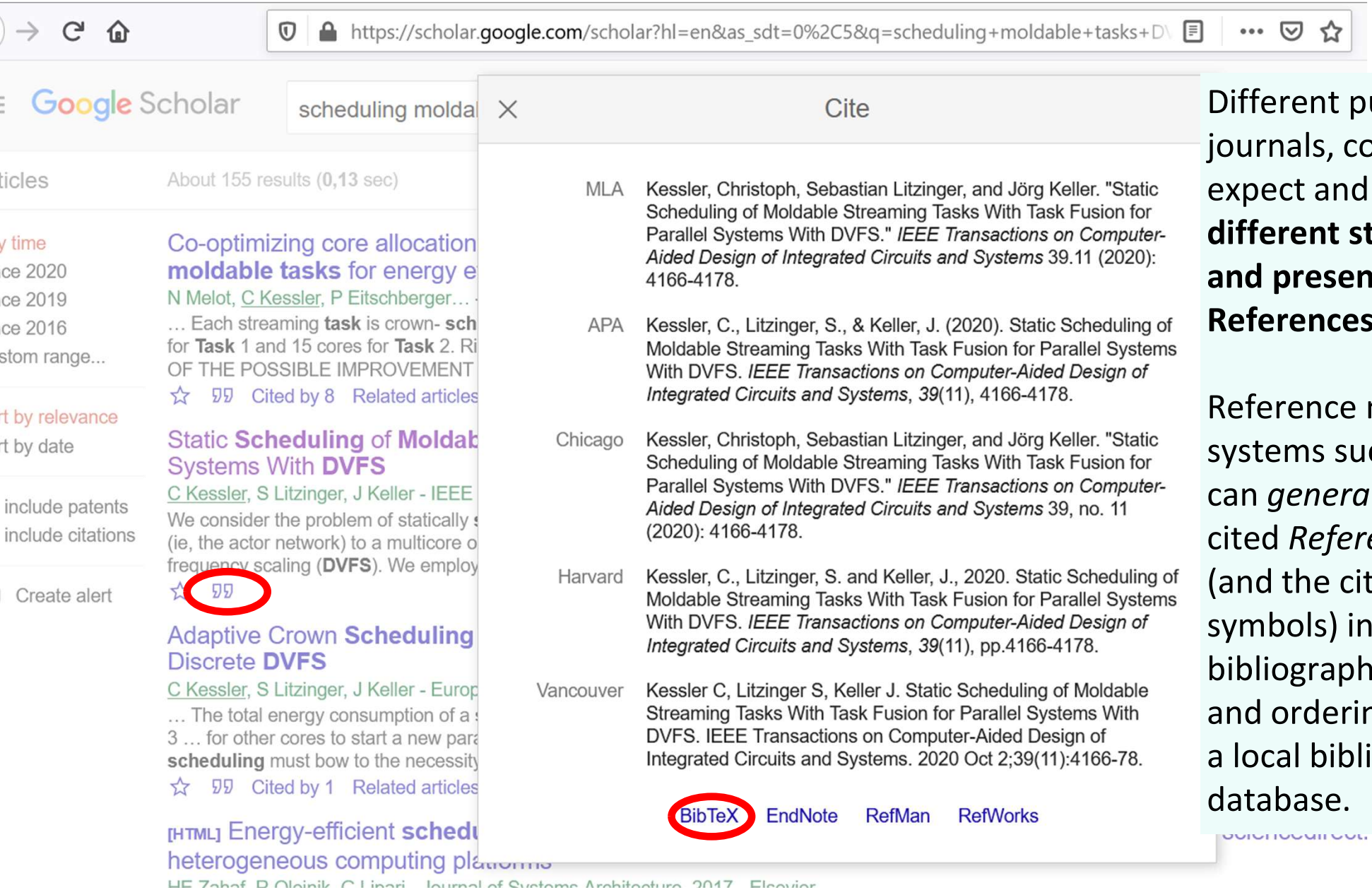
3.8 Refining and Adapting the Model

The QMOOD quality model allows changes to be easily made to the model to address different weightings, other perspectives, and new goals and objectives. At the lowest

→ Use proper quotation + citation if needed, but only quote if really necessary.

[1] J. Bansiya and C. Davis. A hierarchical model for object-oriented design quality assessment. *IEEE Transactions on Software Engineering*, 28(1):4–17, Jan 2002.

Managing references and citations



Different publishers, journals, conferences expect and enforce **different styles of citing and presenting the References**.

Reference management systems such as **BibTEX** can *generate* the list of cited *References* entries (and the citation symbols) in different bibliographic styles and orderings from a local bibliography database.

BiBTEX for LATEX

Unique key (chosen by myself) for my citations, e.g. in:
... see Melot \textit{*et al.*}~**\cite**{Melot2015:TACO} and ...

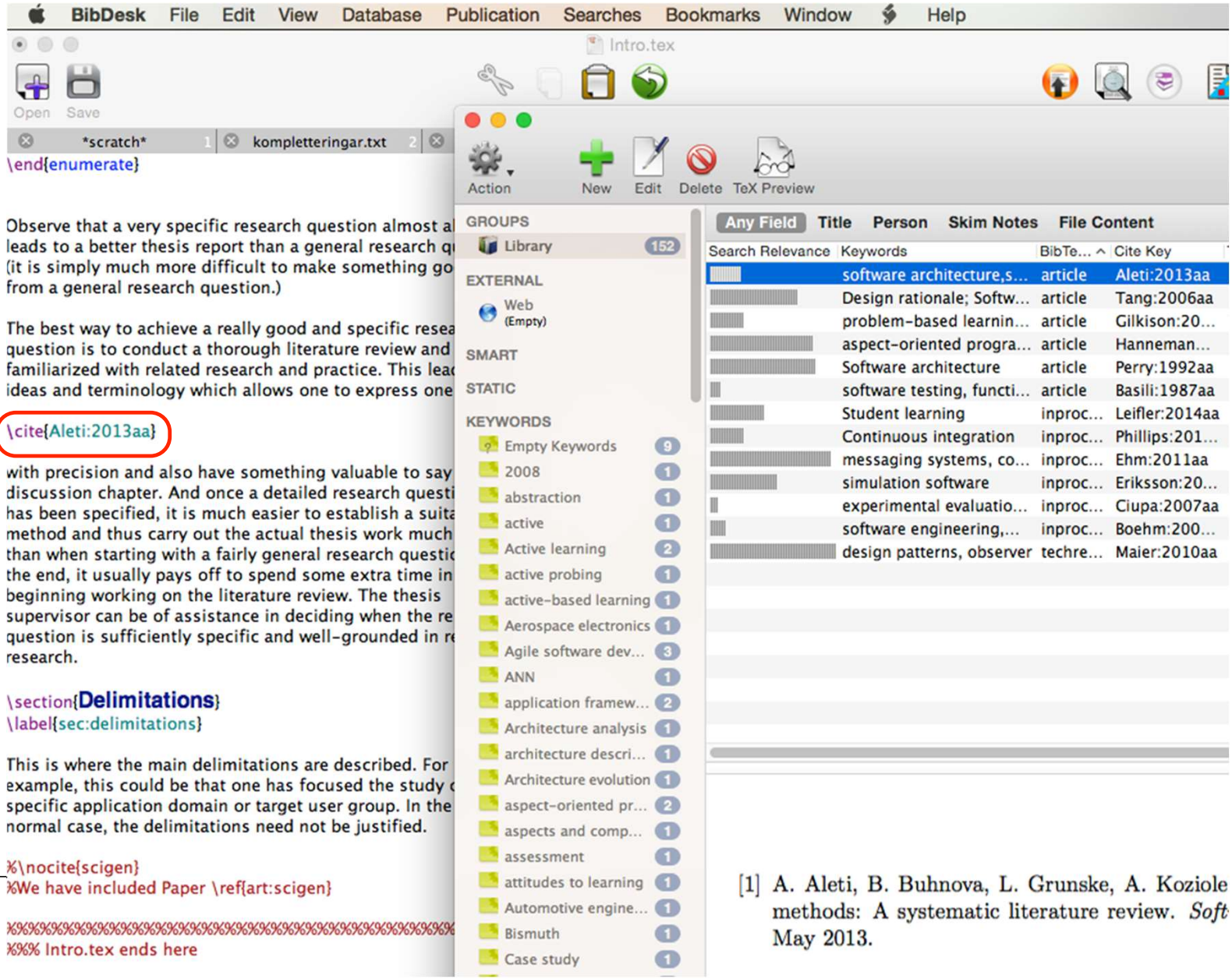
```
@string { TACO = "ACM Transactio                 chitecture and Code Optimization" }

@article{ Melot2015:TACO,
 author  = "Nicolas Melot and Christoph Kessler
            and J{\"o}rg Keller and Patrick Eitschberger",
 title   = "Fast {C}rown Scheduling Heuristics for Energy-Efficient Mapping
            and Scaling of Moldable Streaming Tasks on Manycore Systems",
 journal = TACO,
 volume  = "11",
 number  = "4",
 pages   = "62:1--62:24",
 month   = jan,
 year    = "2015”,
 annote = {See also Melot’s PhD thesis of 2017. Min Energy given throughput.
           Manycore CPU core allocation to moldable parallel actor tasks is
           restricted to a hierarchically decomposition of the set of cores
           (sizes: powers of 2). Combines core allocation, mapping and DVFS.
           Fast heuristics and integer linear programming solution.}
}
```

My internal notes (not part of generated References entry)

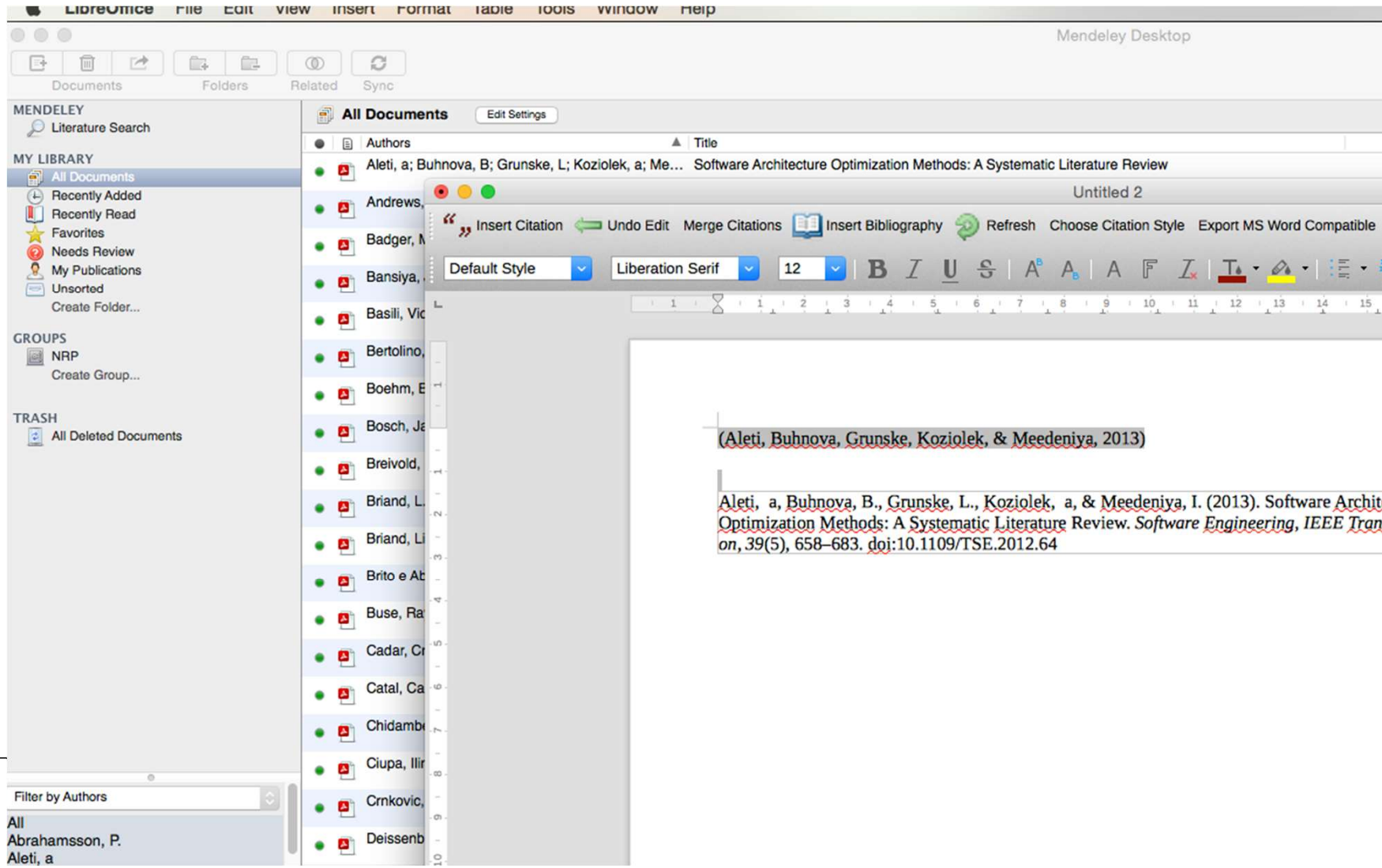
BibDesk

- BibTeX frontend
- Open-source
- MacOS



Mendeley

System
for reference
management
and sharing



LINKÖPINGS UNIVERSITET

Writing about what you have read

- Keep a **copy** (hard or soft) of all papers etc. that you have read
- Take **notes** of what you have read
  - Maybe in the `annote={...}` fields of the entries in your BibTeX file(s)
  - Summarize the main insights *with your own words*
- Consider what needs to be in your thesis.
  Do not write everything you have read into your thesis.
  Remember to keep a strong connection to your main method/results.

Discussing and Comparing Related Work

- **Classify** related approaches and **compare** their properties, assumptions and results with each other and with yours
- Hint: Make a **feature synopsis table**!
  - Shows immediately where / how your solution differs from / outperforms the competitors
  - Helps classifying and grouping the related work description into sections
  - Great to have in a summary section in the Related Work chapter
  - Great to have on a slide in the final presentation

Example taken from: W. Lai *et al.*: A Comparative Study for Single Image Blind Deblurring. CVPR'16, IEEE, 2016

Table 1. Existing and the proposed datasets for performance evaluation of single image blind deblurring algorithms. The proposed data consists of both synthetic and real blurred images. Our real dataset contains real blurred images that cover a wide variety of scenarios. C synthetic dataset includes both uniform and non-uniform blurred images.

Dataset	Levin et al. [17]	Sun et al. [38]	Köhler et al. [13]	Ours (real)	Ours (synthetic)
Synthetic/Real	Synthetic	Synthetic	Real	Real	Synthetic
Blur Model	Uniform	Uniform	Non-uniform	Unknown	Both
Latent Images	4	80	4	100	25
Kernels / Trajectories	8	8	12	100	8
Blurred Images	32	640	48	100	200
Depth variation	No	No	No	Yes	No
Evaluation	PSNR/SSIM	PSNR/SSIM	PSNR	User study	User study

Summary

- Start learning about the subject, then find proper support for your claims.
  Use different sources for learning and as references to support specific claims.
- Know the difference between background literature and related work.
- Do not plagiarize nor violate copyright for images or text.
- There are different types of academic publications and results.
  - Use each type of publication as appropriate.
- Know how to estimate the quality of publications.
  - Read with critical eyes.
  - Your thesis inherits the trust level of the referenced work.
  - You will need to defend your selection in the thesis (“Source criticism”).
- Use proper reference management software when writing your thesis.