Problem-based learning of theoretical computer science

W. Hämäläinen
Department of Computer Science
University of Joensuu

- Problem-based learning method
- Description of the course "Theoretical Foundations of Computer Science" (TFCS) (theory of computability) in problem-based way
- Evaluation of the results

Problem-based learning

Main idea: using problems, queries or puzzles as starting point for learning

In fact variety of approaches, e.g. Ellis & al.:

1. **Problem-based approach**: normal lectures, but problems are used to motivate students and demonstrate the theory.
2. **Guided problem-based learning**: problems are solved in the groups, but also lectures are used to present the fundamental concepts and conceptually most difficult topics.
3. **Full problem-based learning**: problems guide and drive the entire learning experience.

Characteristics

- Acknowledgement of learners' experience.
- Emphasis on students taking responsibility of their own learning.
- Crossing of boundaries between disciplines.
- Focus on the processes of knowledge acquisition rather than the products of such processes.
- Change in staff role from instructor to faciliator.
- Student self- and peer assessment of learning.
- Focus on communication and interpersonal skills.

Advantages

- The students have deeper understanding of the issues
- Better motivation
- Improves communication and cooperation skills
- Improves metacognitive skills like problem solving and ways of thinking.
- Individual learning goals support different learners (also poor students manage well in PBL)
- Gives practise in research and information retrieval

Especially suitable for computer science!

Possible disadvanges?

Critical question: Do the students acquire as good theoretical knowledge as in the traditional education?

Researches:

- Immediately after the course the problem-based learners have better skills but slightly poorer knowledge than traditional learners
- No significant difference in knowledge level, even better results have been reported.
- After some time the theoretical knowledge of PBL learners is at least good as traditional learners – i.e. the PBL learners remember better what they have learnt.

$\leftrightarrow$ **constructivist view of learning:** the students actively construct their knowledge on the basis of their own experience and reflections

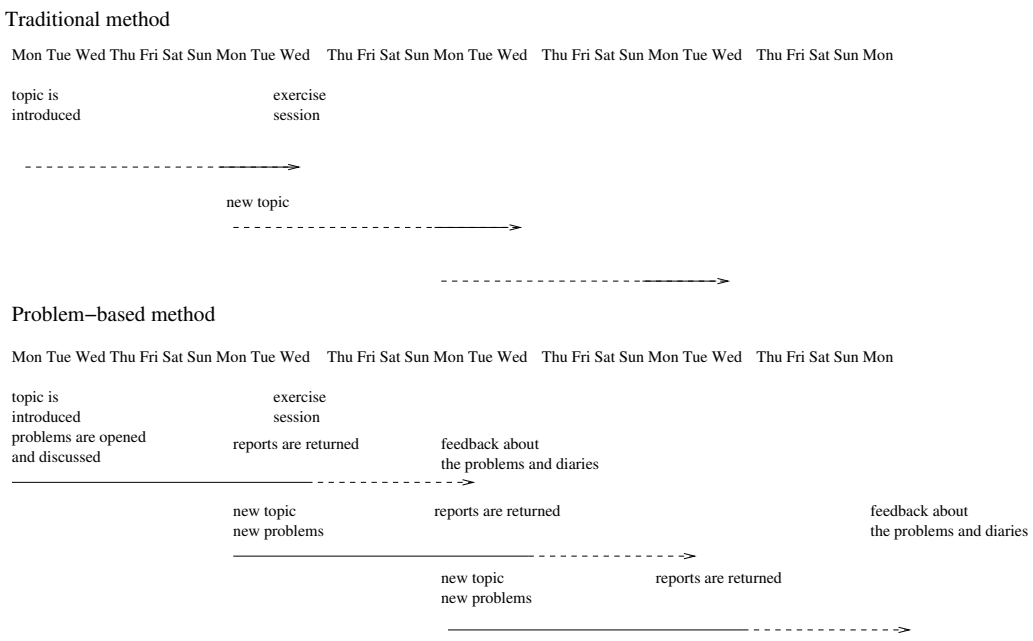

Figure 1: Learning process in traditional and problem-based method. Solid line describes active processing and hash line passive processing.

How to guarantee success?

- Well-structured problems for novice students, open-ended and ill-structured problems for advanced students.
- Learning diary helps to construct overall pictures and supports the learning process.

Experiment with Theoretical Foundations of Computer Science

- 3 cu course about the theory of computability
- 40 h lectures, 20 h exercise sessions
- 75 students began the course (63 problem-based way, 12 traditional)
- 1 lecturer + 2 assistants (teaching exercise sessions)

Arrengements

Contact ”teaching”:

- about 2 hours lectures
- two hours problem solving according to 7-step model
- 2 hours exercise sessions

Homeworks:

- self-study + information collecting
- writing problem reports
- writing learning diary
- solving exercise tasks

Evaluation

Statistics:

	PBL	Trad.	total
began the course	63 $\rightarrow$ 61	12 $\rightarrow$ 14	75
dropped	5 (8 %)	7 (50 %)	12
passed	56 (92 %)	7 (50 %)	63

Grade distribution:

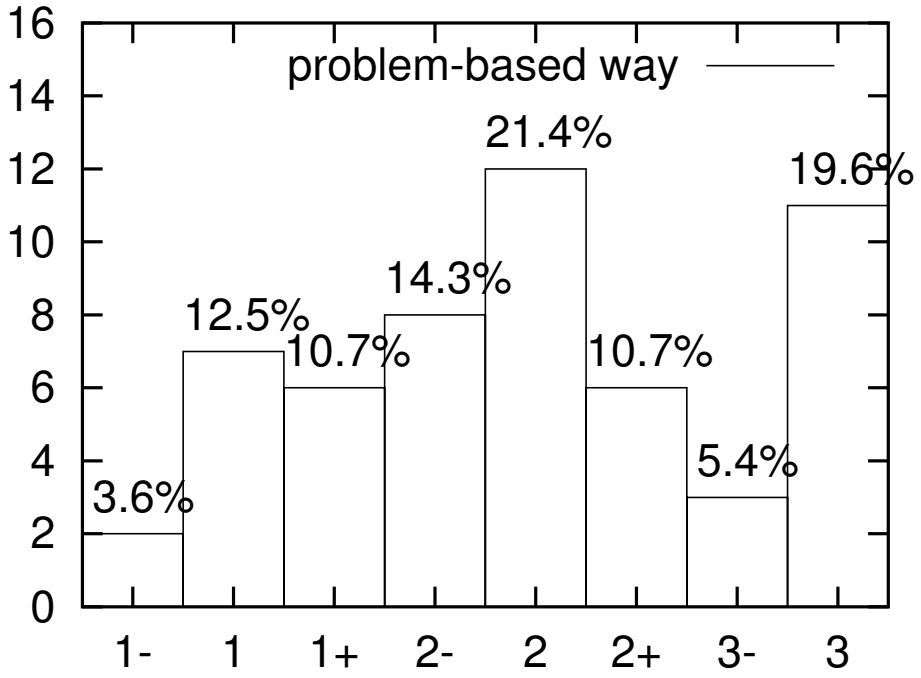

Figure 2: The grade distribution of problem-based learners.

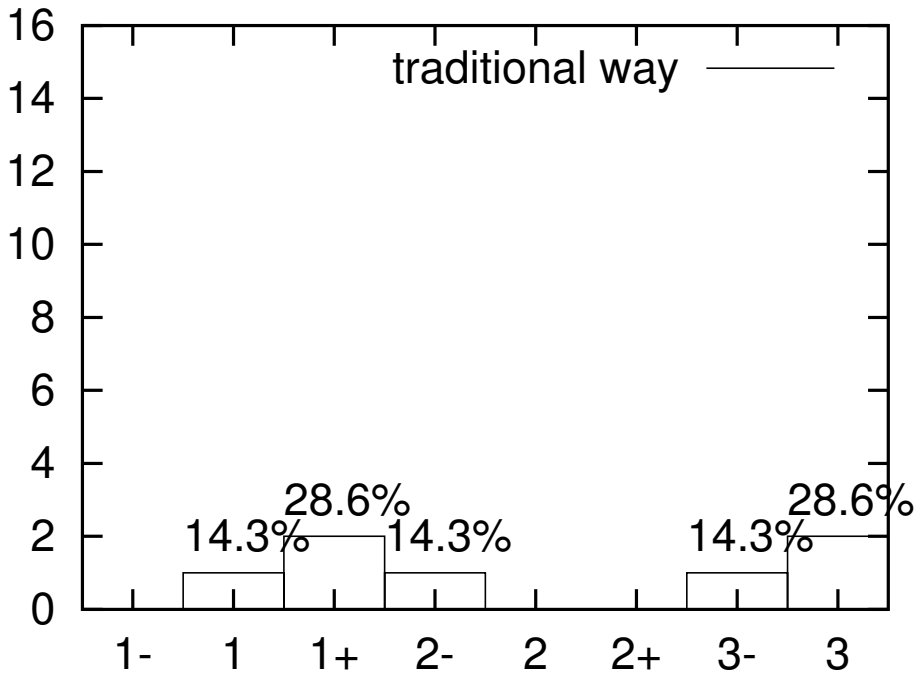

Figure 3: The grade distribution of traditional learners.

Satisfaction with the course

Course evaluation forms:

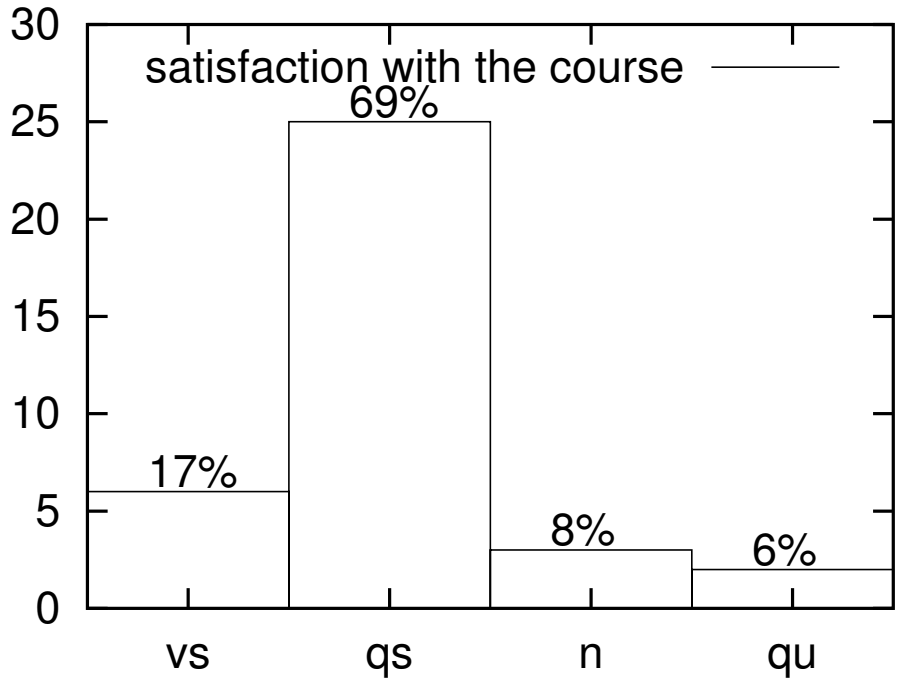

Figure 4: The students' overall satisfaction with the course (vs=very satisfied, qs=quite satisfied, n=normal, qu=quite unsatisfied).

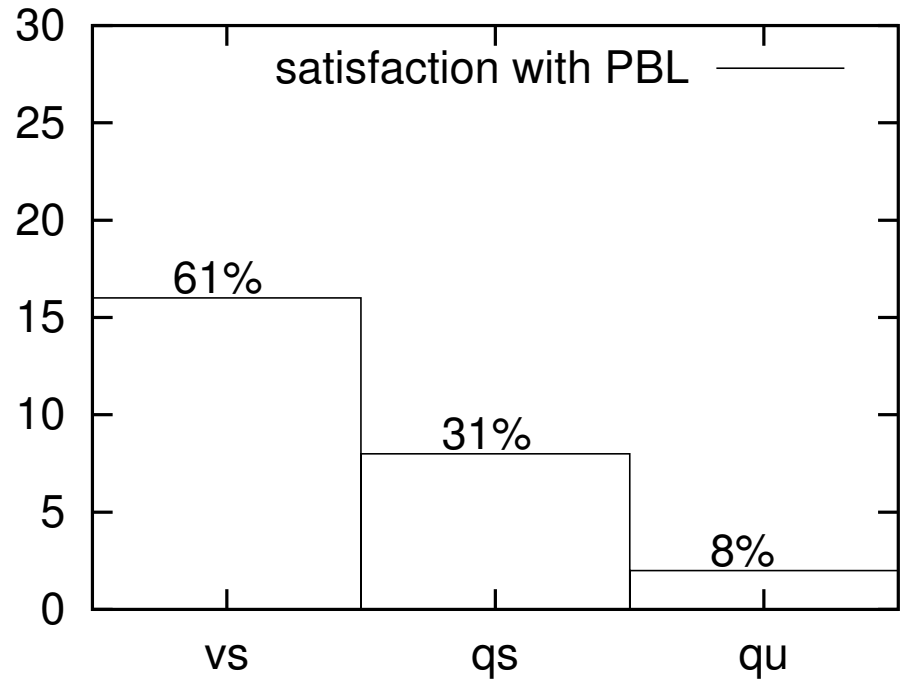

Figure 5: The students' satisfaction with the PBL method (vs=very satisfied, qs=quite satisfied, n=normal, qu=quite unsatisfied).

Other feedback

Students' feedback in learning diaries:

- Students learnt better
- They were better motivated
- They believed they will remember longer
- Foreigner students were more ambivalent

Teachers' feedback:

- More than one tutor should be available in problem sessions! (Suggestion: 1 tutor/20-25 students)
- Assistants' work amount was quite normal, but lecturer's work was enourmous
- Checking problem reports and reading learning diaries take both about 10-15 min/student weekly.
- Teachers agreed that students seemed to learn better and were really motivated.
- Enjoyable also for the teachers!

Conclusions

- A very succesfull experiment (quantity and quality)
- Proved that PBL suits well also for the theoretical courses of cs
- PBL is worth of investing more staff resources
- We recommend warmly!

Seven step method

Phase	Goal
1. Defining unclear concept	The students look for concepts, which are unclear and try to define them.
2. Defining problem	The students discuss about the problem.
3. Brain storming	Students try to construct, test and compare different hypothesis and explanations.
4. Constructing hypotheses	The problem is analyzed carefully by comparing different hypothesis. The ideas are argumented and organized into an intergated whole.
5. Defining learning goals	The students write down their learning goals for self-studying.
6. Self-studying	The students acquiant themselves independently with the topic. In this phase also lectures can be offered to support the self-studying.
7. Sharing the results	The students compare their solutions and try to help each other to understand the topic. Learning goals are checked and final conclusions are drawn.

Problems

Problem	Topic
Case 0	Morse alphabet – Basic concepts
Case 1	Problem solving company – Modelling problems, their difficulty and solvability
Case 2	Search for mail addresses and precompiler for programming language – Regular expressions
Case 3	Coffee automaton – Deterministic finite automata
Case 4	Nondeterministic editor – Nondeterministic finite automata and determinization
Case 5	Roman checking exams – Finite automata vs. regular expressions, ϵ-automata
Case 6	Grandma's rhyme – Pumping Lemma
Case 7	L-systems – idea of grammars
Case 8	Parantheses parsing – Pushdown automaton
Case 9	Arithmetic calculator – LL(1)-grammars, recursive parser
Case 10	General parser – the CYK-algorithm and Chomsky normal form
Extra	Attribute grammar and parser tools
Case 11	Summing machine – basics of Turing machines
Case 12	Library functions for TM's – deeper practice on Turing machines
Case 13	Programming competition – Universal machines and universal languages, solvability proofs
Case 14	Philosophical considerations – Church-Turing theses, limits of computation.
Case 1 revisited	Problem solving company again – Overview of the principles learnt

Table 1: Problem cases and their learning goals.

Classification of the problem setting, goals and methods as open or closed

Problem	Setting	Goal	Methods
Problem 0	**c**	**c**	**o**
Problem 1	**o**	**o**	**o**
Problem 2	**o/c**	**c**	**o**
Problem 3	**c**	**c**	**c**
Problem 4	**o/c**	**o/c**	**o**
Problem 5	**o/c**	**c**	**o**
Problem 6	**o**	**o/c**	**o**
Problem 7	**o**	**o**	**c**
Problem 8	**c**	**c**	**o/c**
Problem 9	**c**	**c**	**o/c**
Problem 10	**c**	**c**	**o/c**
Problem 11	**c**	**c**	**c**
Problem 12	**o**	**o/c**	**c**
Problem 13	**c**	**o**	**o**
Problem 14	**o**	**o**	**o**

Table 2: Classification of the problem setting, goals and methods as open **o** or closed **c**.