

Computer-Supported Negotiation of Course Content[☆]

Roland Hübscher

Department of Information Design and Corporate Communication, Bentley University, 175 Forest Street, Waltham, MA 02452

Abstract

Students learn more effectively with personally meaningful tasks. Thus, students learn more if they have a say in deciding what specific topics and examples are being discussed in class. Naturally, the instructor knows what topics are important to cover in a course and which ones might be optional. Finding the right balance between students' preferences and the instructor's requirements is not so easy and thus may prevent this kind of shared control of the classroom from being realized. This article describes how the instructor's and students' interests can be used to generate a list of course topics that satisfies both parties. However, instead of adding technology to the classroom, technology is used to improve the classroom experiences. Specifically, it is shown how course topics can be assigned to specific students maximizing what is meaningful to the students and satisfies the course parameters as defined by the instructor. This problem can be formulated as a variation of the linear assignment problem and solved with a binary linear program. Results from actual and simulated courses are discussed and generalizations of the topic assignment problem presented.

Key words: Improving classroom teaching, interdisciplinary projects, teaching/learning strategies

1. Introduction

Learning is an active process where students construct their own knowledge and relate it to their previous experiences. Such active learning occurs especially when learners pursue self-driven learning tasks (Collins et al., 1989; Le Cornu and Peters, 2005). Students are often more motivated and engaged in learning if inherently interested in the topic (Blumenfeld, 1992; Petraglia, 1998). The negotiation of content can also result in a potential increase in autonomy of the students (Garrison and Baynton, 1987). Unfortunately, instructional strategies consistent with these ideas are sometimes quite difficult to implement in the classroom. Anybody who has taught knows that covering topics that each student is really interested in is an impossible task. First, students have different interests even if they are studying in the same field. Second, what the instructor believes the students are interested in is frequently no more than just that.

One extreme way to approach this problem is to give the students a free hand in what to learn with the result that they may learn a lot, but not necessarily what the course was meant to teach. A more balanced approach is to let students choose among a pre-screened set of resources (Watts, 1997). Some adaptive hypermedia systems use this approach giving students the opportunity to explore, under the "guiding hand" of the adaptive system (Brusilovsky, 1998), those topics and examples they are interested in most.

A different computational approach is presented in this article. It describes a method that combines the instructor's requirements for the course with the students' preferences. When an instructor designs a course, he or she frequently needs to make choices that are not necessary for pedagogical reasons. Should I use this or that example? Assign this or that paper to read? Have them solve this or that problem? There are multiple good examples, papers to read, and problems to solve. These situations provide a great opportunity to give

[☆]Hübscher, R. (2009). Computer-supported negotiation of course content. *Computers & Education*, 53:726-732.
Email addresses: rhubscher@bentley.edu (Roland Hübscher)

the students a choice so that they end up with a personally more meaningful learning activity. Richards and Lockhardt (1994) found that students participated in a foreign language course more frequently when they could choose what topic to talk about. This is consistent with Brandl (2002) who recommends that students take part in planning the content of a course so that they become active contributors instead of “passive recipients of knowledge.” He differentiates between three types of approaches which are teacher-determined, teacher-facilitated and student-determined. This paper focuses on the teacher-facilitated approach.

Such collaborative classrooms are quite different from the traditional teacher-centric model. Knowledge and authority is shared among teachers and students (Tinzmann et al., 1990). As a result, the teacher is more in the role of a mediator or guide and not the all-knowing oracle. However, giving up this control and developing the appropriate tools can be quite challenging. In this paper, the focus is on the latter. Also, group projects are often assigned to students with quite some flexibility about the specific topic. The goal is that students will be more engaged learners if they choose a topic that will interest them.

If students can, within instructor-defined limits, choose the topics for the whole course, the course can be quite different each time it is taught. This is the specific problem this paper addresses. Although it is assumed that there are more topics than students, the same approach can also be used to assign n topics to n students in such a way that students tend to get a topic they prefer.

As mentioned earlier, the approach presented in this paper can be applied to a variety of choices in the classroom or other situations where it would be advantageous to have people choose some tasks over others. In fact, a student’s preferences for certain learning activities could be replaced with a worker’s qualifications for a certain task. Nevertheless, in the rest of this paper the focus will be on assigning topics to individual students. I will refer to it as the Topic Assignment Problem.

The organization of the paper is as follows. First, the problem is defined and illustrated with a real-world example. Related approaches are discussed next. Then, it is shown how the task of selecting the course topics considering the instructor’s requirements and students’ preferences can be formulated as a variation of a linear assignment problem. Finally, the implications are discussed based on using this approach for several semesters.

2. Choosing Topics for a Course

Students can often be given a choice of what topic to focus on. A trivial example is when the students have to write a paper, and the topic is not too constrained. However, often the situation is slightly more restrictive, such as when the instructor wants to make sure that certain themes are covered to a specified degree. For instance, the instructor may need to choose between many possible articles or examples for certain topics for the students to study and discuss. Letting the students participate in selecting the topics results in learning tasks more meaningful to students.

An example from my own classroom illustrates this situation. In a course on user interfaces, many issues and concepts could be discussed: interfaces for people with disabilities, interfaces for children, interfaces for learners, ambient displays, tangible interfaces, affective computing, and many more. Because there are too many themes to cover in depth in one semester, I mark only some of them (e.g., interfaces for people with disabilities) as required. To each such theme, I assign several sets of papers that address some topic of the theme in some interesting way (e.g., focusing on visually impaired users browsing the World Wide Web, or on motion-impaired users using input devices). Each student will have to focus on a topic covered by one such set of papers, summarize the ideas and lead the discussion in class.

Instructor and students together decide which papers and thus, which topics, are being discussed in each semester. Each student ranks all the given topics on a linear scale, starting with the one that is most interesting to this student, then the second one, and so on. The students have access to the topics for about a week where they can find out which topic matches their interest most. This approach does sometimes result in students picking shorter papers which defeats the purpose of the approach. An alternative approach would be to ask the students to rate all topics according to some characteristics. For instance, for middle-school students active (dynamic, fast-changing) and cool (popular or fashionable) topics were especially interesting (Swarat, 2008).

The instructor marks those themes and topics (there can be several topics per theme) that have to be discussed no matter what the students' preferences are. Furthermore, to avoid all the students being assigned a topic in the same theme, the instructor can restrict how many topics per theme may be assigned by the computational assignment procedure. Of course, the students need not worry about all those instructor-imposed constraints. The goal then is to assign each student a set of papers that is high on his or her list of preferences.

This assignment problem can be formalized as follows. The content of the course is organized as a set of themes and each theme i has a non-empty set t_i of topics. Each of the n student ranks the m topics in a linear list starting with the most preferred topic. The instructor defines for each topic and theme whether it is required. Further, it is assumed that there are at least as many topics as students, i.e., $m \geq n$. Finally, the topics, and thus the sets of papers, are assigned by the algorithm described below to each student, maximizing the students' preferences given the instructor's requirements.

One can also imagine a scenario where instead of requiring that topic A is assigned but not topic B , an instructor might prefer A over B . This can be easily accomplished within the presented framework as will be shown later. Many other extensions could be imagined. Nevertheless, for now the discussion focuses on those constraints and preferences that come up naturally in the context of the real classroom.

3. Existing Approaches

As shown in Section 4.1, the topic assignment problem can be viewed and solved as a variation of the Linear Sum Assignment Problem (LSAP). The LSAP has been used for people assignment problems as early as the 1950s (Votaw and Orden, 1952). Extensions of LSAP have been developed for related but different problems (Burkard, 2002; Toroslu, 2003). In a similar vein, this paper presents an extension of LSAP to support a collaborative classroom where instructor and students share control over the course content.

Recognizing the advantage of letting students choose the topics, the shared-control approach has been used by colleagues and myself using an informal assignment process by hand. However, doing the assignment by hand is exceedingly cumbersome and often leads to suboptimal assignments. The quite efficient Hungarian Algorithm to solve the LSAP (Burgeois and Lasalle, 1971) could, in principle, be done by hand. However, its worst-case complexity is still $O(m^3)$, where m is the number of topics (and normally larger than the number of students). Thus, finding an optimal solution by hand is not realistic, though some people may indeed attempt to find good solutions by hand as I used to in the past. Although finding a good (yet suboptimal) solution may not sound too bad, it has potential negative pedagogical implications. Therefore, optimal solutions are worth finding. Interestingly, the students' informal feedback suggests that explaining to them the general approach of finding an optimal assignment has a positive psychological effect on them in that they believe they indeed got a fair deal when the topics are assigned. These were graduate students, some with a highly technical background, some with a liberal arts background, but all interested in user interface design. It would be interesting to see whether this attitude generalizes to a less computer-friendly student population.

4. Linear Assignment Problems

In this section, the topic assignment problem described in Section 2 is formalized as an extension to the linear sum assignment problem and then solved in the next section with a linear program.

4.1. Linear Sum Assignment Problem

The student preferences of the topic assignment problem can be formalized as a Linear Sum Assignment Problem (LSAP) (Burkard and Çela, 1999). To formulate the instructor's constraints, the LSAP needs to be extended. In its original formulation, a LSAP assigns tasks to machines or workers. Here, we talk about assigning topics to students.

Each student i is assigned a topic $\phi(i)$, i.e., ϕ is a mapping from students to topics. The original formulation of LSAPs assumes that the number of topics m and of students n is the same, whereas for the

topic assignment problem, this condition is relaxed and $n \leq m$ is assumed. An $n \times m$ matrix a can be defined which can be viewed as the graph connecting students i and topics j assigned to them.

$$a_{ij} = \begin{cases} 1 & \text{if } j = \phi(i) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Next, the assignment constraints can be defined. First, we make sure each student gets exactly one topic.

$$\sum_{i=1}^n a_{ij} = 1 \quad \text{for all } j = 1, 2, \dots, m \quad (2)$$

The next constraint assures that each topic is picked at most once. Since in the regular LSAP $m = n$, LSAP requires it to be equal to 1 instead of ≤ 1 .

$$\sum_{j=1}^n a_{ij} \leq 1 \quad \text{for all } i = 1, 2, \dots, n \quad (3)$$

Next, a $n \times m$ cost matrix c is defined where c_{ij} is the cost of assigning topic j to student i . Given the assignment function ϕ we want to minimize the total cost, i.e.,

$$\min \sum_{i=1}^n c_{i\phi(i)} \quad (4)$$

The cost needs to be defined so that better assignments have lower cost. An obvious way to do this is to rank all the topics for each student and use the rank as cost. If the rank of paper j for student i is r_{ij} , then the cost matrix c can be simply defined as $c_{ij} = r_{ij}$ for all $i = 1, 2, \dots, n$ and $j = 1, 2, \dots, m$. A variations of this cost function where worse ranks are punished more heavily than just linearly could be defined with $c_{ij} = r_{ij}^2$.

It is easy to see that the current formulation is a LSAP. The case where $m \geq n$ can be converted to the one where $m = n$ by adding $m - n$ students with zero cost for being assigned any of the topics, i.e., $c_{ij} = 0$ for $i = n + 1, n + 2, \dots, m$ and $j = 1, 2, \dots, m$.

However, the instructor's constraints on which topics and themes must be assigned have not yet been formulated. Let S be the set of all required themes and T be the set of all required topics. Themes and topics can be required by the instructor independently, i.e., a required theme may or may not have required topics, and a required topic may or may not be in a required theme. Therefore, no further requirements need to be imposed on S and T .

First, for each required theme k , at least one of its topics t_k needs to be assigned.

$$\sum_{i=1}^n \sum_{j \in t_k} a_{ij} \geq 1 \quad \text{for all } k \in S \quad (5)$$

Second, each required topic in T needs to be assigned.

$$\sum_{i=1}^n a_{ij} = 1 \quad \text{for all } j \in T \quad (6)$$

Finally, if we want to have a relatively even emphasis on the themes, we can require that no theme has more than h selected topics. I use $h = 2$ in my classes.

$$\sum_{i=1}^n \sum_{j \in t_k} a_{ij} \leq h \quad \text{for all themes } k \quad (7)$$

The topic assignment problem is now defined. Equations 2 and 3 define the assignment constraints, equation 4 defines the optimization criterion describing the students' preferences, and equations 5, 6, and 7 implement the instructor's requirements.

4.2. Linear Program

An LSAP can be solved by many efficient algorithms (Martello et al., 1984). The worst-case complexity of the best algorithm is $O(m^3)$ where m is the number of topics. However, since we have added more constraints, and we solve relatively small problems, it is easier to formulate the problem simply as a binary linear program instead of adapting a specialized algorithm. However, solving a binary linear program is NP-hard Karp (1972), no algorithm with polynomial complexity is known. Still, all linear programs reported in the results section of this paper were solved on a laptop in negligible time for classes under 30 students and up to 2.3 seconds for classes up to 100 students.

A linear program describes a problem in terms of linear equations and inequalities and a linear optimization criterion. Assume we have n real-valued variables x_1, x_2, \dots, x_n , each of which is constrained to be non-negative, i.e., $x_i \geq 0$ for $i = 1, 2, \dots, n$. There are m linear equality or inequality constraints over the x_i , each of the form:

$$\begin{aligned} a_{j1}x_1 + a_{j2}x_2 + \dots + a_{jn}x_n &= b, \\ a_{j1}x_1 + a_{j2}x_2 + \dots + a_{jn}x_n &\leq b, \text{ or} \\ a_{j1}x_1 + a_{j2}x_2 + \dots + a_{jn}x_n &\geq b \end{aligned}$$

Given these constraints, we wish to find values for the x_i that minimize (or maximize) the value of the objective function

$$c + d_1x_1 + \dots + d_nx_n$$

Such problems can be solved efficiently in polynomial time (Karmarkar, 1984), however, the real-valued variables x_i can only occur in linear form.

The formulation of the modified LSAP can be used without further modifications. All we need is add constraints making explicit that the a_{ij} 's are either 0 or 1. Adding these constraints transforms the linear program into an NP-hard binary linear program.

$$\begin{aligned} \sum_{i=1}^n a_{ij} &= 1 && \text{for all } j = 1, 2, \dots, m \\ \sum_{j=1}^m a_{ij} &\leq 1 && \text{for all } i = 1, 2, \dots, n \\ \sum_{i=1}^n \sum_{j \in t_k} a_{ij} &\geq 1 && \text{for all } k \in S \\ \sum_{i=1}^n a_{ij} &= 1 && \text{for all } j \in T \\ \sum_{i=1}^n \sum_{j \in t_k} a_{ij} &\leq h && \text{for all themes } k \\ a_{ij} &\in \{0, 1\} && \text{for all } i = 1, 2, \dots, n \text{ and } j = 1, 2, \dots, m \end{aligned} \tag{8}$$

The optimization criteria needs to be slightly rewritten using the matrix a defined by equation 1.

$$\min \sum_{i=1}^n \sum_{j=1}^m a_{ij} c_{ij} \tag{9}$$

Earlier it was mentioned that instructors could also express preferences and not just constraints. For instance, topics could be given a weight expressing the instructor's preference for them. Let $w_j > 0$ be the weight for topic j where a larger value implies a higher preference. Then, the minimization criterion could be simply rewritten as

$$\min \sum_{i=1}^n \sum_{j=1}^m \frac{a_{ij} c_{ij}}{w_j} \tag{10}$$

5. Results

The linear program has been used in the planning of several courses. The implementation takes as input the themes, topics, instructor's requirements and preferences for each student, and generates a human-readable output via a \LaTeX file (Mittelbach et al., 2004) displaying paper assignments and a schedule for the whole semester. The appendix of this article provides an example implementation of the linear program.

The linear program was applied to real student preferences and instructor constraints. The data for the three classes A , B and C are shown in Table 1. The numbers show how many students were assigned a

Table 1: Results with real student preferences from three classes A , B , and C . The numbers in the columns are how many students had a topic assigned at that rank.

Rank	A	B	C
1	14	11	15
2	2	5	5
3	2	0	1
4	0	2	1
≥ 5	0	0	0
Total	18	18	22

paper ranked first, second, etc. A rank of one means that the student was assigned his or her most preferred topic, a rank equal to two implies that the second-most preferred topic was assigned to him or her, and so on. Approximately 89% of the student were assigned one of their two most-preferred topics, and none was assigned a topic that was below rank four. The reaction of the students to their assignment was always very positive and seems to be reinforced by letting them know that the topic assignment is optimal.

The results from simulated classes show a similar picture. The classes were generated randomly with the number of students $n = 15 \dots 50$ and the number of themes between $n/2$ and $2n/3$. Each theme had between two and six topics. One third of the themes were required and the preferences of the students were random. Assigning the topics in 100 classes resulted in 65.8% of the assigned topics were ranked first and 89.6% were either ranked first or second. No topic below rank five was assigned.

In the presented results, the students' interests were reasonably evenly distributed. This was even more so in the case for the random assignments. However, student preferences can be construed in such a way that even a mathematically optimal solution is not satisfactory. Assume that each of the n students has exactly the same preferences for their top m topics. The best possible assignment results in one topic each at ranks 1 to m leaving quite a few students unhappy.

6. Conclusions

Frequently, a good teaching method requires more effort from the instructor. Sometimes, it seem very difficult to achieve the goal no matter the time one is willing to put into course preparation. Such a situation has been the negotiation of course content between instructor and students. This paper addresses such a situation and provides an elegant solution.

Motivated by the need for assigning students topics (or readings, examples, problems to solve, and so on) they are inherently interested in, a method was developed solving this rather complex computational task. The solution consists of a formulation of the pedagogical needs: Students have interests stated as preferences and instructors have constraints that assure a pedagogically sound selection of topics. The actual solution can be found with a rather simple linear program that efficiently produces good results.

Explaining to the students that their preferences will be used to assign the papers optimally given the parameters set by the instructors has made a big difference. When I used to do it by hand, the paper assignments were suboptimal and challenged by some students. Given that it was done by a person, it was perceived as being somewhat subjective. The reaction to the topic assignments computed with the linear program has always been very positive, and no student has ever shown a dislike about what was assigned to them.

The tool to assign topics to students based on instructor requirements and student preferences has in practice always returned excellent results. The requirements can be expressed in a relatively fine-grained way by listing what topics are available and which ones must be assigned to a student. The students order the topics based on which topic they would like to get assigned. However, in the future, this may be replaced by a more indirect approach where students are asked to evaluate the topics according to a few criteria possibly resulting in a pedagogically improved topic assignment.

Acknowledgements

I would like to thank Teresa Hübscher-Younger and Thom Baguley for their detailed and constructive comments on the manuscript.

A. Implementation

An example implementation of the solution developed in this article is described in this section. The linear program is reformulated using the GNU MathProg modeling language, which is a subset of AMPL,¹ because this language is close to the mathematical formulation used in this article. Linear programs formulated with the MathProg modeling language can be solved, for instance, with the mixed integer linear program solver `lpsolve`² or the GNU Linear Programming Kit (GLPK).³

A toy example with four students and two themes with three topics each is used. Although the model below is written in MathProg, a similar formulation can be used in general programming languages like R, Java or Python.⁴ The toy example is defined as follows:

```
Theme: Interfaces for people with disabilities
  Topics: Lep04, *Hwa03, Fra00
*Theme: Ambient displays
  Topics: Mat06, Pou06, Vog04
Students:
  Jim [Mat06 Fra00 Hwa03 Vog04 Pou06 Lep04]
  Mary [Mat06 Hwa03 Fra00 Lep04]
  Jane [Fra00 Hwa03 Pou06]
  John []
```

The starred topic and theme are required. The topics are given as references to articles in journals and conference proceedings. The preferences of the students are listed in order of decreasing preference. Apparently, John has no preference, that is, he (dis)likes all the topics the same.

The MathProg formulation consists of several parts. First, the parameters and the variable to be computed are specified. Then the optimization criterion and the constraints are defined. Finally, the example-specific data are assigned to the parameters.

First, the sets of students S , topics T and themes E are specified.

```
set S;
set T;
set E;
```

Next, the parameter for the rank r is specified.

```
param r{s in S, t in T};
```

If the value of `theme[e,t]` is 1, t is a topic in theme e . The `binary` keyword implies that the parameter's values are either 0 or 1.

```
param theme{e in E, t in T}, binary;
```

If a topic t or theme e is required, the value of `reqTopic[t]` or `reqTheme[e]` is set to 1, respectively.

¹Additional information about AMPL can be found at <http://www.ampl.com>.

²Lpsolve is available from <http://lpsolve.sourceforge.net/> and can be used as a standalone program or from programming languages such as R, Java or Python.

³GLPK is available from <http://www.gnu.org/software/glpk/>.

⁴The MathProg code described here as well as an implementation in Python can be downloaded from <http://roland.hubscher.org/software/>.

```

param reqTopic{t in T}, binary;
param reqTheme{e in E}, binary;

```

Next, the variable to be computed is defined. If $a[s,t]$ is 1, student s is assigned topic t . This definition corresponds to equation 1.

```

var a{s in S, t in T}, binary;

```

The optimization criteria is to minimize the sum of the ranks of the assigned topics as described by equation 9.

```

minimize cost: sum{s in S, t in T} a[s,t] * r[s,t];

```

The constraints are defined next. Constraint `oneTopicPerStudent` requires that one topic needs to be assigned to each student. It implements equation 2. The abbreviation `s.t.` stands for ‘subject to,’ `{s in S}` states that the constraint holds for all students s , and the constraint is that the number of assigned topics for each s is 1.

```

s.t. oneTopicPerStudent{s in S}: sum{t in T} a[s,t] = 1;

```

A topic can be assigned at most once. This constraints implements equation 3.

```

s.t. topicAtMostOnce{t in T}: sum{s in S} a[s,t] <= 1;

```

A topic from a required theme must be assigned. This constraints implements equation 5.

```

s.t. requiredTheme{e in E}:
    sum{s in S,t in T} a[s,t] * theme[e,t] >= reqTheme[e];

```

Required topics must be assigned. This constraints implements equation 6.

```

s.t. requiredTopic{t in T}: sum{s in S} a[s,t] >= reqTopic[t];

```

No more than two topics per theme should be assigned. This constraints implements equation 7.

```

s.t. noMoreThan2PerTheme{e in E}:
    sum{t in T,s in S} theme[e,t] * a[s,t] <= 2;

```

Finally, solve the linear problem and print out the solution, i.e., the assignments computed in variable a as well as the rank r for each assignment.

```

solve;
printf "\nSOLUTION\n";
printf{s in S,t in T}: if a[s,t] then "%s -> %s\n" else "", s,t,r[s,t];
printf "\n";

```

This concludes the definition of the general topic assignment problem. Next, the actual data needs to be assigned to the parameters and the sets S (students), T (topics) and E (themes).

```

data;

```

```

set S := Jim Mary Jane John;

```

```

set T := Lep04 Hwa03 Fra00 Mat06 Pou06 Vog04;

```

```

set E := IntDisab Ambient;

```

```

param theme : Lep04 Hwa03 Fra00 Mat06 Pou06 Vog04 :=
    IntDisab  1    1    1    0    0    0
    Ambient   0    0    0    1    1    1;

```

```

param reqTopic :=
    Lep04 0   Hwa03 1   Fra00 0   Mat06 0   Pou06 0   Vog04 0;

param reqTheme :=
    IntDisab 0
    Ambient  1;

param r :   Lep04 Hwa03 Fra00 Mat06 Pou06 Vog04 :=
    Jim    5     2     1     0     4     3
    Mary   3     1     2     0     4     4
    Jane   3     1     0     3     2     3
    John   0     0     0     0     0     0;

end;

```

The output of this program when run with GLPK shows that Jim, Jane and John got there first choice and Mary her second. Since John did not state a preference, any assignment results in an optimal assignment for him.

SOLUTION

```

Jim -> Mat06 [0]
Mary -> Hwa03 [1]
Jane -> Fra00 [0]
John -> Vog04 [0]

```

References

- Blumenfeld, P. C., 1992. Classroom learning and motivation: Clarifying and expanding goal theory. *Journal of Educational Psychology* 84 (3), 272–281.
- Brandl, K., 2002. Integrating internet-based reading materials into the foreign language curriculum: From teacher- to student-centered approaches. *Language Learning & Technology* 6 (3), 87–107.
- Brusilovsky, P., 1998. Adaptive educational systems on the world-wide-web: A review of available technologies. In: 4th International Conference in Intelligent Tutoring Systems. San Antonio, TX.
- Burgeois, F., Lasalle, J.-C., 1971. An extension of the Munkres algorithm for the assignment problem to rectangular matrices. *Communications of the ACM* 14 (12), 802–804.
- Burkard, R. E., 2002. Selected topics on assignment problems. *Discrete Appl. Math.* 123 (1-3), 257–302.
- Burkhard, R. E., Çela, E., 1999. Linear assignment problems and extensions. In: Du, D.-Z., Pardalos, P. M. (Eds.), *Handbook of Combinatorial Optimization*. Kluwer Academic Publishers, pp. 75–149.
- Collins, A., Brown, J. S., Newman, S. E., 1989. Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics. In: Resnick, L. B. (Ed.), *Knowing, Learning, and Instruction*. Erlbaum, Hillsdale, NJ, pp. 453–494.
- Garrison, D. R., Baynton, M., 1987. Beyond independence in distance education: The concept of control. *American Journal of Distance Education* 1 (3), 3–15.
- Karmarkar, N., 1984. A new polynomial-time algorithm for linear programming. *Combinatorica* 4, 373–395.
- Karp, R., 1972. Complexity of Computer Computations. Plenum, Ch. Reducibility Among Combinatorial Problems, pp. 85–103.
- Le Cornu, R., Peters, J., 2005. Towards constructivist classrooms: the role of the reflective teacher. *Journal of Educational Enquiry* 6 (1), 50–64.
- Martello, S., Pulleyblank, W. R., Toth, P., de Werra, D., 1984. Balanced optimization problems. *Operations Research Letters* 3, 275–278.
- Mittelbach, F., Goossens, M., Braams, J., Carlisle, D., Rowley, C., 2004. *The L^AT_EX Companion: Tools and Techniques for Computer Typesetting*, 2nd Edition. Addison-Wesley Professional.
- Petraglia, J., 1998. *Reality by Design: The Rhetoric and Technology of Authenticity in Education*. Lawrence Erlbaum Associates, Mahwah, NJ.
- Richards, J. C., Lockhardt, C., 1994. *Reflective Teaching in Second Language Classrooms*. Cambridge University Press.
- Swarat, S., 2008. What makes a topic interesting? A conceptual and methodological exploration of the underlying dimensions of topic interest. *Electronic Journal of Science Education* 12 (2), 66–93.
- Tinzmann, M., Jones, B., Fennimore, T., J. Bakker, C. F., Pierce, J., 1990. What is the collaborative classroom? URL [http://aeonline.coe.utk.edu/PDFramework/Instructional Approaches/Collaborative.pdf](http://aeonline.coe.utk.edu/PDFramework/Instructional%20Approaches/Collaborative.pdf)
- Toroslu, I. H., 2003. Personnel assignment problem with hierarchical ordering constraints. *Comput. Ind. Eng.* 45 (3), 493–510.

- Votaw, D. F., Orden, A., 1952. The personnel assignment problem. In: Orden, A., Goldstein, L. (Eds.), Proceedings of the Symposium of Linear Inequalities and Programming, Project SCOOP, No. 10. Planning Research Division, Director of Management Analysis Service, Comptroller, USAF, pp. 155–163.
- Watts, N., 1997. A learner-based design model for interactive multimedia language learning packages. System 25 (1), 1–8.