Maastricht University

Perspectives on problem solving and instruction

Citation for published version (APA):

Van Merrienboer, J. J. G. (2013). Perspectives on problem solving and instruction. *Computers & Education*, *64*, 153-160. https://doi.org/10.1016/j.compedu.2012.11.025

Document status and date:
Published: 01/05/2013

DOI:
10.1016/j.compedu.2012.11.025

Document Version:
Publisher's PDF, also known as Version of record

Document license:
Taverne

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:

www.umlib.nl/taverne-license

Take down policy
If you believe that this document breaches copyright please contact us at:

repository@maastrichtuniversity.nl

providing details and we will investigate your claim.

Download date: 25 Apr. 2024

Computers & Education 64 (2013) 153–160

Contents lists available at SciVerse ScienceDirect

Computers & Education

journal homepage: www.elsevier.com/locate/compedu

Perspectives on problem solving and instruction

Jeroen J.G. van Merriënboer

Maastricht University, The Netherlands

ARTICLE INFO

Article history:
Received 21 June 2012
Received in revised form
19 October 2012
Accepted 14 November 2012

Keywords:
Problem solving
Instructional design
Complex learning

ABSTRACT

Most educators claim that problem solving is important, but they take very different perspective on it and there is little agreement on how it should be taught. This article aims to sort out the different perspectives and discusses problem solving as a goal, a method, and a skill. As a goal, problem solving should not be limited to well-structured problem solving but be extended to real-life problem solving. As a method, problem solving has clear limitations for novice learners; providing ample support to learners is of utmost importance for helping them to develop problem-solving skills. As a skill, problem solving should not be seen as something that only occurs in the early phases of a process of expertise development but as a process that develops in parallel in System 1 and System 2. The four-component instructional design model (4C/ID) is briefly discussed as an approach that is fully consistent with the conceptualization described in this article and as a preliminary answer to the question how problem solving is best taught.

© 2013 Published by Elsevier Ltd.

1. Introduction

All life is problem solving (Popper, 1999). In everyday and professional contexts, everyone frequently solves problems. When we wake up, we must decide what clothes to wear. In schools, teachers must deal with management problems in their classroom and students must determine how much time to spend on different types of school work. In professional life, workers are required to organize complex projects, deal with interpersonal conflicts, and develop innovative products. And in the evening, we must decide whether we relax on the couch, play a sport, or visit the theater. What all these situations have in common is that there is an unknown entity (e.g., the clothes to wear) in some situation, that is, a difference between the current state (wearing your pajama) and a desirable goal state (wearing appropriate clothes). Jonassen (2000) makes the important point that finding or solving for the unknown in such real-life problem situations not only has some intellectual value, but also a social or cultural value. Thus, solving problems is not only an integral part of life but also helps people feel valuable.

Whereas most educators regard problem solving as critical for life, there is yet little agreement on how problem solving should be taught in schools, other educational institutions and the workplace. One reason is that problem solving is an extremely complex cognitive process about which little is known. We should more deeply understand the breadth and complexity of problem-solving processes in order to be able to effectively engage and support learners in them. Moreover, the scientific discussion on problem solving in education is a Tower of Babel, making it even more difficult to reach some consensus on how to teach problem solving. For example, some educators reserve the term problem solving for the use of cognitive methods that can be applied in any domain, while others stress the importance of domain knowledge in problem solving. Some focus on the importance of problem solving as an educational goal that can best be reached by teaching known solutions, while others advocate the use of problem solving as an educational method. And some see problem solving as an early phase in the process of expertise development, while others see problem solving as one important aspect of fully developed – reflective – expertise.

The main goal of this article is twofold. First, it aims to sort out the chaos and distinguish the most important perspectives on problem solving. Second, it aims to provide a preliminary answer to the question how real-life problem solving is best taught. What makes this article original is that it not only highlights the various perspectives of problem solving but also critically analyzes how these different perspectives are conceptually interrelated with each other.

The structure of this article is as follows. The first section discusses different perspectives on problem solving as an educational *goal*. A distinction is made between weak problem-solving methods, strong problem-solving methods, knowledge-based problem-solving

E-mail address: j.vanmerrienboer@maastrichtuniversity.nl.

0360-1315/$ – see front matter © 2013 Published by Elsevier Ltd.
http://dx.doi.org/10.1016/j.compedu.2012.11.025

methods, and a combination of strong and knowledge-based methods in real-life problem solving. The second section discusses different perspectives on problem solving as an educational *method*. It will be argued that problem solving might be a goal but is not an appropriate educational method for novice learners, and that effective educational methods should carefully and gradually help learners move toward this goal. The third section discusses different perspectives on problem solving as a *skill*. A distinction is made between phase models, which typically see problem solving as an early phase in the development of expertise, and System 1/System 2 models, which view problem solving as an important feature of two interacting systems. The fourth section briefly discusses the four-component instructional design model (4C/ID; Van Merriënboer, 1997; Van Merriënboer & Kirschner, 2013) as a preliminary answer to the question how real-life problem solving is best taught. The fifth and final section provides the main conclusions and directions for further research.

2. Problem solving as an educational goal

Educational researchers and practitioners greatly differ in their definition of problem solving. A first definition refers to the use of *weak methods*, which can be used to solve unfamiliar, new problems in *any* domain. There are many of them, such as hill-climbing, means-ends analysis, generate-and-test, heuristic search, subgoal decomposition, hypothesize-and-match, constraint satisfaction, and pure forward search (e.g., Newell & Simon, 1972). An example of one of a number of domain-general methods for the weak method of hill-climbing is: If the goal is to transform the current state into a goal state, then set as subgoals to (i) find the largest difference between the current state and the goal state, (ii) find an operator to eliminate that difference, and (iii) convert the state that results from the application of this operator to the goal state. Thus, this domain-general method tries to reach a goal state by looking for available operators that may eliminate the difference between the current state and the desired goal state. When you are traveling in a foreign country, this method may help you identify the means of transportation (e.g., bus, train, aircraft) that best helps you move from your current to your final destination. Although weak methods allow one to solve problems in any unfamiliar domain, it is highly questionable whether the teaching of such methods should be a primary goal of education. First, weak methods will only be effective if the information they operate upon is correct. They are not able to generate acceptable behavior if they operate on incorrect information from the outside world or from the problem solver's memory. Second, the costs related to the use of weak methods are extremely high. The problem-solving process is slow, will often be unsuccessful, and the load on working memory is exceptionally high. The latter is true because the interpretation of declarative information requires continuous retrieval of this information from memory or the outside world, and this information must be held active in working memory. Third, the general assumption is that weak methods are innate (Anderson, 1993) and that it may be impossible to teach them because they are 'wired in' the human cognitive architecture. Indeed, attempts to teach domain-general problem solving have typically been unsuccessful (Sweller, Clark, & Kirschner, 2010).

A second definition of problem solving refers to *strong methods*, which can be used to solve specific problems in a particular domain. Strong methods are typically described as highly-domain specific *if-then* rules that generate a solution to well-structured problems, that is, problems that present all elements of the problem to the learner, require the application of a limited number of rules or procedures, and have knowable, comprehensible solutions (Frederiksen, 1984; Jonassen, 1997). Many typical school tasks are well-structured problems that can be solved by strong methods: Procedures for addition, subtraction, division and multiplication in arithmetic; formulas for doing computations in physics and science; grammatical rules for the conjugation of verbs, and so forth. When people can recognize problems as belonging to a particular class and have the applicable rules available, they can use their specific knowledge to solve these problems (i.e., *same* use of the same knowledge). Moreover, strong methods are algorithmic, meaning that their correct application under appropriate conditions guarantees that the problem is solved: Correctly applying the procedure for doing addition, for example, will always yield the correct answer independent of the numbers that are added. After extensive amounts of practice, the application of rules or procedures may become fully automatic so that learners respond to the problem "what is the sum of 14 and 3?" with the answer 17, without the need to consciously apply the procedure anymore. Thus, strong methods may eventually be applied very fast and with low demands on working memory, but they are highly inflexible because they are only applicable to specific problems. Although most educators will agree that strong methods should be taught in education, many of them would not classify the application of strong methods as problem solving but rather as "just performing a routine". But it is equally justified to call it the most extreme, efficient type of problem solving one can think of.

A third definition of problem solving refers to *knowledge-based methods*, which can be situated between weak and strong methods. They may help to find an acceptable solution for ill-structured problems, that is, problems that contain unknown elements, have multiple acceptable solutions (or no solution at all), possess multiple criteria for evaluating solutions, and often require learners to make judgments. A very important contribution of David Jonassen (e.g., 1997) is that he was one of the first educational researchers to stress the importance of using ill-structured problems in education and training and much of his work investigated instructional methods for teaching ill-structured problem solving. This definition of problem solving refers to the interpretation of domain knowledge and/or previously encountered cases to come up with possible solution steps. It stresses the importance of deep understanding of a domain in order to effectively solve problems in it. If problem solvers have a good understanding of how things are named and interrelated in a domain (i.e., *conceptual* models), how things work and affect each other in a domain (i.e., *causal* models), and how things are built or organized in a domain (i.e., *structural* models), they may use this general knowledge to restructure a given problem situation and to infer tentative solutions for the problem. Alternatively, they can use their memories of previously encountered cases as an analogy to come up with possible solutions (analogical problem solving; Gick & Holyoak, 1980; Jonassen, 2002). In addition, they may also use domain-specific cognitive strategies that help them approach problems in a systematic fashion and apply rules-of-thumb that help them to successfully complete each phase in a systematic problem-solving process. Knowledge-based methods thus allow one to solve problems in a particular domain of learning but do not guarantee that an acceptable solution is reached: They are heuristic rather than algorithmic. Although knowledge-based methods are much more efficient than weak methods, they are yet slow, error-prone and effort-demanding in comparison with strong methods. The reason is that they refer to the *different* use of the same knowledge, which requires conscious interpretation by the problem solver. This makes knowledge-based problem-solving methods much more flexible than strong methods, which refer to the *same* use of the same knowledge. Most educationalists will argue that knowledge-based methods should be taught in education and, indeed, knowledge is typically taught in the hope that learners will eventually use it to solve problems.

A fourth and final definition combines the perspectives on well-structured and ill-structured problem solving and pertains to *real-life problem solving* (Van Merriënboer, 1997). Whereas the distinction between ill-structured and well-structured problem solving is valid from

a theoretical point of view, real-life problems will almost without exception require a mix of ill-structured and well-structured problem solving and, in addition, the cognitive processes responsible for ill-structured problem solving (i.e., knowledge-based methods) and well-structured problem solving (i.e., strong methods) must be coordinated by the task performer. Real-life problem solving refers to the kind of problem solving one encounters in writing essays, conducting research, diagnosing patients in medicine, controlling air traffic, engineering software, and so forth. When proficient task performers solve problems in these domains they will typically apply knowledge-based methods to find an acceptable solution. But they are only able to do so because they can apply strong methods for the routine aspects of performing the task, "... making available controlled-processing resources for the novel aspects of problem solving" (Frederiksen, 1984, p. 365). In modern education, authentic learning tasks that are based on real-life tasks (e.g., projects, case studies, simulations) are increasingly seen as the driving force for teaching and learning because they are instrumental in helping learners to integrate their knowledge, skills and attitudes (Merrill, 2002; Van Merriënboer & Kirschner, 2013). In this educational context, problem solving always refers to the simultaneous use of strong methods for routine aspects of performance and knowledge-based methods for non-routine aspects of performance (e.g., reasoning, decision making). The remainder of this article will therefore focus on real-life problem solving.

3. Problem solving as an educational method

The previous section argued that problem solving is broadly seen as a highly desirable educational goal, but, how should it then be taught? A popular belief is that good educational methods should mimic the processes they aim to develop. The argumentation is that learning to solve problems is of utmost importance and that in order to achieve this goal we must "thus" use problem solving as an educational method in schools. The use of problem solving as an educational method, however, completely ignores human working memory limitations (Kirschner, Sweller, & Clark, 2006; Sweller, 1988; Van Merriënboer & Sweller, 2005, 2010). Working memory is very limited in duration and in capacity. Information stored in working memory and not rehearsed is lost within 30 s (Baddeley, 1992, 2000) and the capacity of working memory is limited to only a small number of 7 plus or minus 2 elements (Miller, 1956) or even 4 plus or minus 1 element (Cowan, 2001). When actively processing rather than merely storing information as during problem solving, the number of items that can be processed may only be 2 or 3 depending on the nature of the processing required (Sweller, van Merriënboer, & Paas, 1998).

The interactions between working memory and long-term memory are even more important than the direct processing limitations (Sweller, 2004). The limitations of working memory only apply to new, yet to be learned information that has not been stored in long-term memory. When dealing with previously learned information stored in long-term memory, the limitations disappear because an experienced problem solver has constructed cognitive schemas in long-term memory that can be used to solve new problems. These schemas are handled as *one* element in working memory. But in the absence of cognitive schemas, as is the case for novice problem solvers, problem solving is only possible thanks to the weak methods described above; these methods require the student to consider differences between the goal state and the given state of the problem, and to search blindly for solution steps to reduce those differences. Problem solving through weak methods is the only way of attaining a problem goal in the absence of useful cognitive schemas. This process is exceptionally expensive in terms of working memory capacity, because the problem solver must continually hold and process in working memory the current problem state, the goal state, the relations between goal state and problem state, the solution steps that could reduce the differences between the two states, and any subgoals along the way. More importantly, this process bears no relation whatsoever to schema construction processes concerned with learning to recognize problem states and their associated solution steps, that is, with learning to solve problems. Learning to solve problems and problem solving are thus two very different and incompatible processes!

Several alternatives to conventional problem solving have been devised to teach problem solving in more efficient and effective ways, including the use of goal-free problems, worked examples, and completion problems (see Van Merriënboer & Sweller, 2005, for an overview of alternative methods). Goal-free problems do not permit problem solvers to extract differences between a current problem state and a goal state because no goal is specified. In order to solve goal-free problems, a learner considers the problem state encountered and finds a solution step that can be applied, yielding a new problem state for which the learner can find another solution step, et cetera. With regard to working memory, a goal-free strategy requires nothing more than each problem state and any solution step that can be applied to that state, which drastically reduces working memory load. It is precisely this combination that is required for the construction of cognitive schemas. Studying worked examples also eliminates means-ends search and reduces cognitive load. In contrast to conventional problems, worked examples focus students' attention on relevant problem states and associated solution steps, enabling them to construct useful cognitive schemas. An alternative for worked examples is provided by completion problems, because they overcome the disadvantage of worked examples that they do not force learners to carefully study them. Completion problems are problems for which a given state, a goal state, and a partial solution are provided to learners who must complete the partial solution. Like worked examples, completion problems decrease cognitive load, but unlike worked examples, they force learners to study them because they otherwise will not be able to complete the solution correctly.

There is overwhelming evidence that goal-free problems, worked examples, and completion problems are much more effective than conventional problem solving to teach problem solving and reach transfer of learning, that is, the ability to solve new problems in a domain (see Sweller, Ayres, & Kalyuga, 2011). This is not only true for well-structured but also for ill-structured problems, although it is then necessary to pay explicit attention to the knowledge-based methods (i.e., interpretation of mental models, conscious use of cognitive strategies) that help an expert problem solver to obtain an acceptable solution. This is reached by so-called modeling examples. They confront learners with professionals performing the complex task, who are simultaneously explaining the processes used to reach an acceptable solution (Van Gog, Paas, & van Merriënboer, 2006, 2008). Thinking-aloud during the problem-solving process has proven a very helpful technique for bringing the hidden mental problem-solving processes of the professional into the open (Van Gog, Paas, van Merriënboer, & Witte, 2005). A more specific approach in perceptual domains (e.g., medical diagnosis, air traffic control) is the use of eye movement modeling examples, which not only disclose the cognitive processes but also the perceptual processes of an expert so that for each moment in time the learner can see what the expert is looking at and in what sequence (Van Gog, Jarodzka, Scheiter, Gerjets, & Paas, 2009). By studying the modeling examples, learners can get a clear impression of how experts use their knowledge to identify promising solution steps, the problem-solving phases they go through, and the rules-of-thumb they use to overcome impasses and complete each phase successfully.

Although the value of problem solving as an educational method is highly overestimated, this is not to say that problem solving cannot be used as an educational method at all. This is due to the *expertise reversal effect* (Kalyuga, Ayres, Chandler, & Sweller, 2003). Research on expertise reversal indicates that highly effective educational methods for novice learners (e.g., goal-free problems, worked out examples, modeling examples, completion tasks) can lose their effectiveness and even have negative effects on learning when used with more experienced learners. Then, the support that is provided by examples may not be necessary or may even be detrimental to learning because more experienced learners have already acquired the cognitive schemas that guide their problem-solving processes; they have their own, proven personal and/or idiosyncratic ways of working. These cognitive schemas may interfere with the examples or other means of support provided to them. Rather than risking conflict between the experienced learners' available cognitive schemas and the support provided by the instruction, it is then preferable to eliminate the support. Fading-guidance strategies (Renkl & Atkinson, 2003) take the expertise reversal effect into account and sustain the gradual development of novice problem solvers into more experienced problem solvers. An effective fading-guidance strategy is, for example, the 'completion strategy' (Van Merriënboer, 1997; Van Merriënboer & Kirschner, 2013), which uses completion problems as a bridge between worked examples or modeling examples (i.e., completion problems with a full given solution and, for modeling examples, articulation of the process to reach this solution) and conventional problems (i.e., completion problems without a given solution). In the completion strategy, learners start with the study of worked examples or modeling examples, then complete increasingly larger parts of given partial solutions, and only independently solve conventional problems after lengthy and substantial practice.

Concluding, for novice learners' problem solving is *not* an effective educational method although it might be a goal. Effective educational methods should carefully and gradually help learners move toward this goal. First, such methods should help learners to gain some knowledge about the learning domain, because problems can only be efficiently solved thanks to things you already know. Second, such methods should provide support during the problem-solving process, and only decrease support as learners gain more experience. Thus, they should systematically sustain the development of problem-solving *skills* over time.

4. Problem solving as a skill

When problem solving is conceptualized as a skill, it is seen as something that develops over time as a function of practice. Within the skills perspective, two types of models can be distinguished: Phase models and System 1/System 2 models. *Phase models* typically link problem solving to one or more phases in a process of expertise development or skill acquisition. Dreyfus and Dreyfus (1980), for example, describe the development of expertise as a progression through five phases: Novice, competence, proficiency, expertise and mastery. They see problem solving as characteristic of the novice phase, and argue that "... skill in its minimal form is produced by following abstract formal rules [cf. *weak methods*], but that only experience with concrete cases can account for higher levels of performance" (p. 5). As another example, J. R. Anderson's ACT-theory (Adaptive Control of Thought; 1983, 1993) describes the acquisition of complex cognitive skills as a progression through three phases, which resemble the cognitive phase, associative phase, and autonomous phase originally described by Fitts and Posner (1967). In the first phase, learners use weak problem-solving methods to generate initial solutions in new problem-solving situations. In the second phase, a process called knowledge compilation makes the transition from slow, controlled processing to more automatic processing possible. Domain-specific cognitive rules are created from the initial solutions by incorporating domain-specific knowledge in cognitive rules (a subprocess called *proceduralization*) and by combining rules that consistently follow each other while performing particular tasks (a subprocess called *composition*). In the third phase, cognitive rules accumulate strength each time they are successfully applied. After extensive practice, this process of strengthening may eventually make the performance of problem-solving skills fully automatic.

According to phase models, an expert would simply be described as someone who has automated most of his or her task performance. This might be true in domains such as music, sports or even well-structured school tasks, but in many other complex domains experts do not only differ from novices in that they have automated many routine aspects of tasks, but their deep understanding of the domain also allows them to interpret new problem situations in more general terms, to monitor and to reflect on the quality of own performance, and to detect and correct errors. Thus, expertise not only shows the ability to perform routine aspects of problems highly automatically, but *also* the ability to solve non-routine aspects of problems by knowledge-based methods, to evaluate the validity of reached solutions, and to switch between problem approaches when necessary. These two different aspects of expert behavior are better reflected in *system models* than in phase models. System models make a distinction between automatic processing, which is fast, unconscious, inflexible and intuitive because it uses mental shortcuts (System 1), and controlled processing, which is slow, conscious, flexible and effortful (System 2; Kahneman, 2011; see also Schneider & Shiffrin, 1977; Shiffrin & Schneider, 1977). System 1 and System 2 work in parallel and interact with each other, especially, System 2 can be employed to monitor the quality of the answers provided by System 1 and if it is convinced that our intuition is wrong, then it is capable of correcting or overriding the automatic judgments. Novices and experts differ from each other in *both* System 1 and System 2 processing.

Table 1 describes the implications of System 1–System 2 models for learning real-life problem solving (Van Merriënboer, 1997). First, it is clear that practice aimed at the development of such skills must make an appeal on the development of *both* System 1 and System 2 processing, and that learners must also learn to coordinate both types of processing. In other words, practice must aim at the development of routine aspects of problem-solving behavior as well as the development of non-routine aspects of problem-solving behavior, such as reasoning (i.e., use domain knowledge to infer tentative problem solutions) and conscious decision making (i.e., use cognitive strategies to approach problems in a systematic fashion). For a novice learner, those aspects that need to be developed into System 1 behaviors are called *recurrent* skills (Van Merriënboer, 1997); they are treated as being consistent from problem situation to problem situation. Critical to the development of recurrent skills is *repetitive practice*. For example, after vast amounts of repetitive practice journalists may have become expert touch-typists because they developed cognitive rules that drive particular actions under particular circumstances; their finger movements are directly driven by their thoughts regardless of whether the text they are writing is an article on science, politics or history. In contrast, those aspects that need to be developed into non-routine, System 2 behaviors are called *non-recurrent* skills; they are treated as being different from problem situation to problem situation. Critical to the development of non-recurrent skills is *variability of practice* (Paas & van Merriënboer, 1994), meaning that learners should practice on problems differing on all dimensions on which they also differ from each

Table 1
Implications of System 1/System 2 models for learning real-life problem solving methods.

Cognitive principles	Learning principles
Real-life problem-solving always relies on a mix of System 1 and System 2 processing.	- Well-designed practice should make an appeal on *both* routine aspects (System 1) and non-routine aspects (System 2). - Practice should be repetitive for routine aspects (System 1). - Practice should be varied for non-routine aspects (System 2).
System 1 processing (*strong methods*) not only develops through repetitive practice, but initially requires how-to instructions during practice and immediate feedback.	- Information relevant for routine aspects should take the form of how-to instructions, is best presented just-in-time and promotes knowledge compilation. - Feedback on routine aspects should be immediate, signifies errors and provides hints for how to continue.
System 2 processing (*knowledge-based methods*) not only develops through variability in practice, but initially requires elaboration of relevant knowledge and feedback-by-reflection.	- Information relevant for non-routine aspects should take the form of domain models and systematic approaches to problem solving and promote elaboration. - Feedback on non-routine aspect should be delayed and promote reflection by asking learners to compare own problem-solving processes and solutions with others' problem-solving processes and solutions.
System 1 processing develops much slower than System 2 processing.	- Routine aspects that need to be developed to a very high level of automaticity require additional repetitive practice.

other in the real world. For example, only after writing many different types of articles the same journalists may have become expert writers because they developed domain knowledge and cognitive strategies allowing them to make good decisions on the content and structure of new articles.

Information provision for recurrent and non-recurrent aspects of real-life problem-solving skills is also fundamentally different, because different learning processes are relevant to System 1 and System 2. A central learning process for the development of System 1 processing is knowledge compilation, the process through which new knowledge is converted into highly specific cognitive rules. The timing of information presentation is important because the new information must be active in working memory at the precise moment that it is needed to carry out the task, it will thus often take the form of *how-to* instructions that are presented just-in-time, precisely when learners need them to perform the task they are working on. The same principle applies for the provision of immediate feedback. In order to facilitate knowledge compilation, feedback should be given immediately after making an error, inform the learner why there was an error, and give a hint for how to continue without simply saying what the correct action is (Balzer, Doherty, & O'Connor, 1989).

In contrast, the central learning process for the development of System 2 processing is *elaboration*, establishing meaningful relationships both between newly presented information elements and between these newly presented elements and what is already known by the learner (i.e., his or her prior knowledge). Elaboration of new information, such as domain models that help learners to develop mental models of a learning domain and/or systematic approaches to problem solving that help learners develop cognitive strategies for approaching problems in a domain, will typically take place before learners start to practice. This process of elaboration yields rich cognitive schemas that relate many elements to many other elements. Such schemas allow for deep understanding and increase the availability and accessibility of task-related knowledge in long-term memory. Then, it will best help learners perform and learn to perform the non-recurrent aspects of real-life problem-solving skills. The same principle applies for the provision of so-called *cognitive* feedback. In order to facilitate elaboration, feedback should be delayed until learners have completed one or more problem-solving tasks. It should promote reflection-on-action and stimulate learners to compare and contrast their own problem-solving processes and solutions with the problem-solving processes and solutions of others, such as expert models or peers (Butler & Winne, 1995).

Finally, System 1 processing develops much slower than System 2 processing. Fully automatic, intuitive System 1 processing only occurs after a long-lasting process of *strengthening*, where cognitive rules accumulate strength each time they are successfully applied by the learner. The Power Law of Practice (Newell & Rosenbloom, 1981) explains why the development of fully automatic System 1 behaviors is extremely slow. The law predicts that the log of the time to complete a response will be a linear function of the log of the number of successful executions of that particular response. For example, if the time needed to add two digits decreased from 3 s to 2 s over the first 100 practice items, it will take 1.6 s after 1000 items, 1.3 s after 10,000 items, and about 1 s to add two digits after 100,000 trials. For learning real-life problem-solving tasks, it thus seems advisable to practice recurrent and non-recurrent aspects of the task not only together, so that learners can learn to coordinate those aspects, but to provide additional practice for recurrent aspects that need to become fully automatic, such as when children drill-and-practice multiplication tables or when musicians practice specific musical scales.

5. Implications for problem solving instruction

The principles described in Table 1 provide the basis for the four-component instructional design model (4C/ID-model; Van Merriënboer, 1997; Van Merriënboer & Kirschner, 2013). This model assumes that environments for teaching real-life problem-solving skills can always be described as being built from four components: (1) learning tasks, (2) procedural information, (3) supportive information, and (4) part-task practice.

Well-designed *learning tasks* are based on real-life tasks and thus make on appeal on both recurrent and non-recurrent aspects of problem-solving skills. According to the 4C/ID-model, a sequence of learning tasks provides the backbone of an educational program (see Fig. 1; learning tasks are represented as boxes). It is clearly not possible to begin an educational program with very complex learning tasks because this would yield excessive cognitive load for the learners (Van Merriënboer & Sweller, 2005, 2010). The solution is to let learners

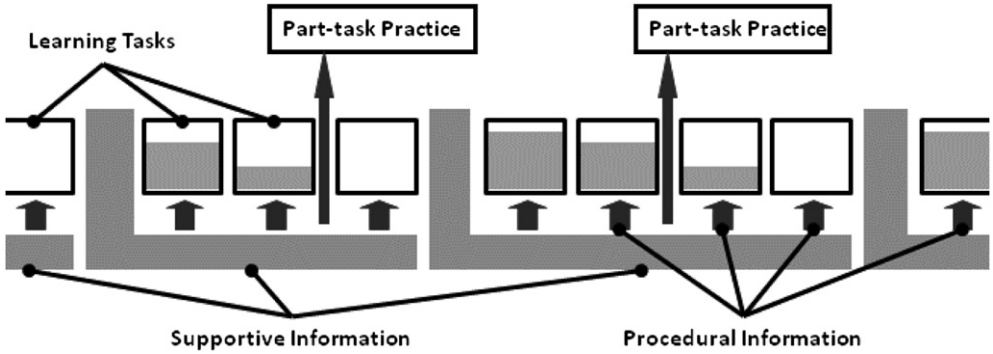

Fig. 1. Schematic representation of the four components in an educational program.

start work on relatively simple but yet "whole" learning tasks and gradually progress toward more complex tasks. Categories of learning tasks, each representing a version of the problem-solving tasks with a particular complexity, are called task classes (see the clusters of tasks in Fig. 1). Learning tasks within a particular task class are always equivalent in the sense that they can be performed based on the same body of knowledge (i.e., mental models and cognitive strategies). When learners begin to work on a new, more complex task class, it is essential that they receive instructional support (in Fig. 1, indicated by the gray in the boxes). This support diminishes in a process of scaffolding as learners acquire more expertise. There are many different ways to support learners' work on learning tasks, but a highly effective fading-guidance strategy already described above is the completion strategy, where learners first study worked examples and/or modeling examples, then complete increasingly larger parts of incomplete examples, and eventually solve problems on their own (Renkl & Atkinson, 2003; Van Merriënboer, 1990).

Suitable media for learning tasks must allow learners to work on those tasks and will usually take the form of a real or simulated task environment, including tools and objects necessary for carrying out the tasks. In some cases, the real task environment such as the future workplace is a suitable setting for learners to perform their learning tasks (i.e., an internship). There may, however, be good reasons not to choose for this option, but rather to choose practice carrying out the learning tasks in a simulated task environment. Particularly in the earlier phases of the learning process (i.e., task classes at the beginning of the educational program), simulated task environments may offer better opportunities for learning than real task environments in that they provide a safe environment where learners can make errors, and where tasks can be provided at an optimal level of complexity and with an optimal level of support. Because learning tasks are based on real-life tasks their psychological fidelity is always high: The cognitive processes needed to perform the learning tasks are similar or equivalent to the cognitive processes needed to perform the real-life tasks. A related design decision concerns the level of *physical* fidelity of the simulated task environment, which is defined as the degree of similarity between the simulated task environment and the real task environment. This may be very low, as for textual problem descriptions of patients presented in a web-based course for medical students. It may be intermediate, as for lifelike simulated characters (i.e., avatars) that can be interviewed in a virtual reality environment. Or it may be very high, as for a full-fledged operating room where medical residents treat a computerized mannequin who reacts just like a real patient.

Procedural information is primarily important for those aspects of real-life problem solving that have been classified as being recurrent. It specifies for learners how to perform routine aspects of learning tasks and preferably takes the form of direct how-to instruction and immediate feedback. In order to facilitate knowledge compilation, it is preferably presented precisely when and where the learners need it for working on the learning tasks (in Fig. 1, it is indicated by the upward pointing arrows connected to individual learning tasks). The traditional media for presenting procedural information are the teacher and all kinds of job aids and learning aids. The teacher's role is to walk around, peer over the learner's shoulder and give directions for performing the routine aspects of learning tasks (e.g., "no – you should hold that instrument like this…", "watch, you should now select this option…"). Job aids may be the posters with frequently used software commands that are hung on the walls of computer classes, quick reference guides adjacent to a piece of machinery, or booklets with instructions on the house-style for interns at a company. In computer-based environments, the presentation of procedural information is often taken over by online job aids and help systems, wizards, and pedagogical agents. Smartphones and tablets are also quickly becoming important tools for presenting procedural information. Such devices are particularly useful for presenting small displays of information that tell learners during task performance what to do in order to perform the routine aspects of the task at hand correctly.

Supportive information is primarily important for those aspects of real-life problem solving classified as being non-recurrent. It explains to learners how a learning domain is organized (by providing conceptual, causal and structural domain models) and how to systematically approach problems in that domain. In order to facilitate elaboration, learners are encouraged to deeply process the new information, in particular by connecting the new information to what they already know. Because supportive information is relevant to all learning tasks at the same level of complexity, it is typically presented before learners start to work on a new task class and kept available for them during their work on this task class (in Fig. 1, this is indicated by the L-shaped shaded areas). A more complex task class requires more supportive information or more embellished supportive information than the preceding, simpler task classes. Traditional media for teaching supportive information are textbooks and teachers. They describe models of a domain and provide descriptions and models of how to systematically approach problem-solving tasks in that domain. Computer-based hypermedia and multimedia systems may take over these functions. Computer-based simulations of conceptual domains are a special category of multimedia in that they offer a highly interactive approach to the presentation of cases where learners can change the settings of particular variables and study the effects of those changes on other variables (De Jong & van Joolingen, 1998). The main goal of such *microworlds* is not to help learners practice real-life problem-solving skills (as is the case in computer-simulated task environments), but to help them construct, through active exploration and experimentation, mental models of how the world is organized and cognitive strategies of how to systematically explore this world.

Finally, *part-task practice* helps learners reach a very high level of automaticity for selected recurrent aspects of real-life problem-solving tasks – it thus explicitly aims at System 1 processing. Part-task practice enables strengthening of cognitive rules and should thus provide huge amounts of repetition. Part-task practice for a particular recurrent aspect should only begin after this aspect has been introduced in meaningful whole learning tasks (see Fig. 1). In this way, the learners are able to see how part-task practice can contribute to whole-task performance. Traditional media for part-task practice include paper-and-pencil for doing small exercises (e.g., simple addition, verb conjugation), skills labs for practicing perceptual-motor skills (e.g., operate machinery, give intravenous injections), and the real task environment. For part-task practice, the computer has also proved its worth in the last decades. Drill-and-practice computer-based training is a successful type of educational software. The computer is sometimes excoriated for its use of drill-and-practice, but most criticism misses the point. Critics contrast drill-and-practice with educational software that focuses on authentic, real-life problem-solving tasks. According to the 4C/ID-model, however, part-task practice never replaces meaningful whole-task practice. It merely complements work on learning tasks and is applied only when the learning tasks themselves cannot provide enough practice to reach the desired level of automaticity for selected recurrent task aspects. If such part-task practice is necessary, the computer is a highly suitable medium because it can make drill-and-practice effective and appealing through the presentation of procedural support, by compressing time so that more exercises can be completed than in real time, by giving knowledge of results and immediate feedback on errors, and by using multiple representations, gaming elements and sound effects.

6. Conclusions

This article aimed to sort out the chaos about the term problem solving and to provide a preliminary answer to the question how real-life problem solving is best taught. It has been argued that problem solving is one of the most important goals of education. But as a goal it should not be limited to well-structured problem solving, but be extended to real-life problem solving including the joint application of strong problem-solving methods and knowledge-based problem-solving methods. As an educational method, problem solving yet has clear limitations for novice learners. Providing support to learners is of utmost importance for helping them to develop problem-solving skills, and in the early phases of learning alternative methods such as goal-free problems, worked examples, modeling examples and completion problems are much more effective than conventional problems. As a skill, problem solving does not need to be seen as something that only occurs in the early phases of a process of expertise development or skill acquisition, but it can and should be seen as a process that develops in parallel in System 1 (development of strong methods for recurrent aspects of real-life problem solving) and System 2 (development of knowledge-based methods for non-recurrent aspects of real-life problem solving). With regard to the teaching of real-life problem solving, the 4C/ID-model was briefly described as an approach that is fully consistent with the conceptualization described in this article (see Van Merriënboer & Kirschner, 2013, for a full description).

The ideas described in this article are indebted to the work of David Jonassen. There is a shared focus on real-life problem solving which he would could call ill-structured problem solving; on the use of authentic problems or tasks as a basis for teaching problem solving; on the importance of a gradual decrease in support and guidance (or, scaffolding) while learning problem solving, and so forth. Yet, Jonassen (2000) would probably argue that the 4C/ID-model is not applicable to teaching all different types of problem solving. In 2000, he presented an extensive typology of 11 different problem types: Logical problems, algorithmic problems, story problems, rule-using problems, decision-making problems, troubleshooting problems, diagnosis-solution problems, strategic performance problems, case analysis problems, design problems, and dilemmas. In line with the principle of "conditions of learning" (Gagné, 1980), which states that different learning outcomes require different educational methods, Jonassen claims that optimal instructions will be different for the different types of problems.

On the one hand, it is certainly to be expected that the *content* of what is taught will be significantly different for different types of problems. For example, learning to solve algorithmic problems may only require the provision of procedural information and not of supportive information. Learning to solve troubleshooting problems may primarily require the provision of causal models because such models allow for reasoning about the functioning and malfunctioning of systems (Jonassen & Ionas, 2008). Learning to solve decision-making problems may primarily require the provision of systematic approaches to problem solving and rules-of-thumb (Jonassen, 2012). And learning to solve dilemmas will probably require the provision of rich conceptual models but not require any part-task practice. Yet, on the other hand, it is an empirical question whether different types of problems require different *approaches* to instructional design, or even require learning environments that cannot be described as being built from the four components. Until now, the 4C/ID-model proved to be useful to design problem-solving instruction for a wide range of different types of problems (Van Merriënboer & Kirschner, 2013). This is not surprising, because all different kinds of problems share the same fundamental characteristic: Task performers are required to find or solve for an unknown that bridges the gap between a given state and a goal state. Nevertheless, future research is needed to further disentangle the requirements to instructional design and learning environments that are posed by the different types of problems as described by Jonassen (2000).

To conclude, it should be pointed out that problem solving is seen as the driving force for student-centered learning in many contemporary learning theories such as 4-Mat (McCarthy, 1996), cognitive apprenticeship (Collins, Brown, & Newman, 1989), constructivist learning environments (Jonassen, 1999), goal-based scenarios (Schank, Berman, & McPerson, 1999), and problem-based learning (Loyens, Kirschner, & Paas, 2011). Although these theories share their focus on problem solving as an important educational goal, a clear conceptualization of what problem solving is, how it develops over time as a function of practice, and how this development process is best supported by different instructional measures typically remains implicit. This article aimed to clarify what the "working ingredients" in such programs are. This is important because, in my view, these programs are essential because "all life is problem solving" and learners must be well prepared for that.

References

Anderson, J. R. (1983). *The architecture of cognition*. Cambridge, MA: Harvard University Press.
Anderson, J. R. (1993). *Rules of the mind*. Hillsdale, NJ: Lawrence Erlbaum Associates.
Baddeley, A. (1992). Working memory. *Science, 255*, 556--559.
Baddeley, A. (2000). The episodic buffer: a new component of working memory? *Trends in Cognitive Sciences, 4*, 417–422.
Balzer, W. K., Doherty, M. E., & O'Connor, R. (1989). Effects of cognitive feedback on performance. *Psychological Bulletin, 106*, 410–433.

Butler, D. L., & Winne, P. H. (1995). Feedback and self-regulated learning: a theoretical synthesis. *Review of Educational Research, 65*, 245–281.
Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: teaching the craft of reading, writing, and mathematics. In L. B. Resnick (Ed.), *Knowing, learning, and instruction: Essays in honor of Robert Glaser* (pp. 453–493). Hillsdale, NJ: Lawrence Erlbaum Associates.
Cowan, N. (2001). The magical number 4 in short-term memory: a reconsideration of mental storage capacity. *Behavioral and Brain Sciences, 24*, 87–114.
De Jong, A. J. M., & van Joolingen, W. R. (1998). Scientific discovery learning with computer simulations of conceptual domains. *Review of Educational Research, 68*, 179–201.
Dreyfus, S. E., & Dreyfus, H. L. (1980). *A five-stage model of the mental activities involved in directed skill acquisition* (Technical report ORC-80-2). Berkeley, CA: Operations Research Center, University of California.
Fitts, P. M., & Posner, M. I. (1967). *Human performance*. Oxford, UK: Brooks and Cole.
Frederiksen, N. (1984). Implications of cognitive theory for instruction in problem solving. *Review of Educational Research, 54*, 363–407.
Gagné, R. M. (1980). *The conditions of learning*. New York: Holt, Rinehart, & Winston.
Gick, M. L., & Holyoak, K. J. (1980). Analogical problem solving. *Cognitive Psychology, 12*, 306–356.
Jonassen, D. H. (1997). Instructional design models for well-structured and ill-structured problem-solving learning outcomes. *Educational Technology Research and Development, 45*, 65–94.
Jonassen, D. H. (1999). Designing constructivist learning environments. In C. M. Reigeluth (Ed.). *Instructional design theories and models: A new paradigm of instructional theory, Vol. 2* (pp. 215–239). Mahwah, NJ: Lawrence Erlbaum Associates.
Jonassen, D. H. (2000). Toward a design theory of problem solving. *Educational Technology Research and Development, 48*, 63–85.
Jonassen, D. H. (2002). Case-based reasoning and instructional design: using stories to support problem solving. *Educational Technology Research and Development, 50*, 65–77.
Jonassen, D. H. (2012). Designing for decision making. *Educational Technology Research and Development, 60*, 341–359.
Jonassen, D. H., & Ionas, I. G. (2008). Designing effective supports for causal reasoning. *Educational Technology Research and Development, 56*, 287–308.
Kahneman, D. (2011). *Thinking, fast and slow*. New York: Farrar, Straus and Giroux.
Kalyuga, S., Ayres, P., Chandler, P., & Sweller, J. (2003). The expertise reversal effect. *Educational Psychologist, 38*(1), 23–31.
Kirschner, P. A., Sweller, J., & Clark, R. E. (2006). Why minimal guidance during instruction does not work: an analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching. *Educational Psychologist, 46*(2), 75–86.
Loyens, S., Kirschner, P. A., & Paas, F. (2011). Problem-based learning. In S. Graham, A. Bus, S. Major, & L. Swanson (Eds.), *Applications to learning and teaching. APA educational psychology handbook, Vol. 3*. Washington, DC: American Psychological Association.
McCarthy, B. (1996). *About learning*. Barrington, IL: Excell Inc.
Merrill, M. D. (2002). First principles of instructional design. *Educational Technology Research and Development, 50*, 43–59.
Miller, G. A. (1956). The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological Review, 63*, 81–97.
Newell, A., & Rosenbloom, P. (1981). Mechanisms of skill acquisition and the law of practice. In J. R. Anderson (Ed.), *Cognitive skills and their acquisition* (pp. 1–55). Hillsdale, NJ: Lawrence Erlbaum Associates.
Newell, A., & Simon, H. A. (1972). *Human problem solving*. Englewood Cliffs, NJ: Prentice-Hall.
Paas, F., & van Merriënboer, J. J. G. (1994). Variability of worked examples and transfer of geometrical problem-solving skills: a cognitive-load approach. *Journal of Educational Psychology, 86*, 122–133.
Popper, K. (1999). *All life is problem solving*. London, UK: Routledge.
Renkl, A., & Atkinson, R. K. (2003). Structuring the transition from example study to problem solving in cognitive skill acquisition: a cognitive load perspective. *Educational Psychologist, 38*, 15–22.
Schank, R. C., Berman, T. R., & MacPerson, K. A. (1999). Learning by doing. In C. M. Reigeluth (Ed.). *Instructional design theories and models: A new paradigm of instructional theory, Vol. 2* (pp. 161–181). Mahwah, NJ: Lawrence Erlbaum Associates.
Schneider, W., & Shiffrin, R. M. (1977). Controlled and automatic human information processing: I. Detection, search, and attention. *Psychological Review, 84*, 1–66.
Shiffrin, R. M., & Schneider, W. (1977). Controlled and automatic human information processing: II. Perceptual learning, automatic attending, and a general theory. *Psychological Review, 84*, 127–190.
Sweller, J. (1988). Cognitive load during problem solving: effects on learning. *Cognitive Science, 12*, 257–285.
Sweller, J. (2004). Instructional design consequences of an analogy between evolution by natural selection and human cognitive architecture. *Instructional Science, 32*, 9–31.
Sweller, J., Ayres, P., & Kalyuga, S. (2011). *Cognitive load theory*. New York: Springer.
Sweller, J., Clark, R. E., & Kirschner, P. A. (2010). Teaching general problem-solving skills is not a substitute for, or a viable addition to, teaching mathematics. *Notices of the AMS, 57*(10), 1303–1304.
Sweller, J., van Merriënboer, J. J. G., & Paas, F. (1998). Cognitive architecture and instructional design. *Educational Psychology Review, 10*, 251–296.
Van Gog, T., Jarodzka, H., Scheiter, K., Gerjets, P., & Paas, F. (2009). Attention guidance during example study via the model's eye movements. *Computers in Human Behavior, 25*, 785–791.
Van Gog, T., Paas, F., & van Merriënboer, J. J. G. (2006). Effects of process-oriented worked examples on troubleshooting transfer performance. *Learning and Instruction, 16*, 154–164.
Van Gog, T., Paas, F., & van Merriënboer, J. J. G. (2008). Effects of studying sequences of process-oriented and product-oriented worked examples on troubleshooting transfer efficiency. *Learning and Instruction, 18*, 211–222.
Van Gog, T., Paas, F., van Merriënboer, J. J. G., & Witte, P. (2005). Uncovering the problem-solving process: cued retrospective reporting versus concurrent and retrospective reporting. *Journal of Experimental Psychology: Applied, 11*, 237–244.
Van Merriënboer, J. J. G. (1990). Strategies for programming instruction in high school: program completion vs. program generation. *Journal of Educational Computing Research, 6*, 265–285.
Van Merriënboer, J. J. G. (1997). *Training complex cognitive skills*. Englewood Cliffs, NJ: Educational Technology Publications.
Van Merriënboer, J. J. G., & Kirschner, P. A. (2013). *Ten steps to complex learning* (2nd rev. ed.). New York: Routledge.
Van Merriënboer, J. J. G., & Sweller, J. (2005). Cognitive load theory and complex learning: recent developments and future directions. *Educational Psychology Review, 17*, 147–177.
Van Merriënboer, J. J. G., & Sweller, J. (2010). Cognitive load theory in health professional education: design principles and strategies. *Medical Education, 44*, 85–93.