Available online at www.sciencedirect.com

ScienceDirect

Interacting with Computers 20 (2008) 64–96

Interacting with Computers

www.elsevier.com/locate/intcom

Pedagogy and usability in interactive algorithm visualizations: Designing and evaluating CIspace

Saleema Amershi *, Giuseppe Carenini, Cristina Conati, Alan K. Mackworth, David Poole

Department of Computer Science, University of British Columbia, 201-2366 Main Mall, Vancouver, BC, Canada V6T 1Z4

Received 10 February 2006; received in revised form 25 April 2007; accepted 1 August 2007
Available online 18 September 2007

Abstract

Interactive algorithm visualizations (AVs) are powerful tools for teaching and learning concepts that are difficult to describe with static media alone. However, while countless AVs exist, their widespread adoption by the academic community has not occurred due to usability problems and mixed results of pedagogical effectiveness reported in the AV and education literature. This paper presents our experiences designing and evaluating CIspace, a set of interactive AVs for demonstrating fundamental Artificial Intelligence algorithms. In particular, we first review related work on AVs and theories of learning. Then, from this literature, we extract and compile a taxonomy of goals for designing interactive AVs that address key pedagogical and usability limitations of existing AVs. We advocate that differentiating between goals and design features that implement these goals will help designers of AVs make more informed choices, especially considering the abundance of often conflicting and inconsistent design recommendations in the AV literature. We also describe and present the results of a range of evaluations that we have conducted on CIspace that include semi-formal usability studies, usability surveys from actual students using CIspace as a course resource, and formal user studies designed to assess the pedagogical effectiveness of CIspace in terms of both knowledge gain and user preference. Our main results show that (i) studying with our interactive AVs is at least as effective at increasing student knowledge as studying with carefully designed paper-based materials; (ii) students like using our interactive AVs more than studying with the paper-based materials; (iii) students use both our interactive AVs and paper-based materials in practice although they are divided when forced to choose between them; (iv) students find our interactive AVs generally easy to use and useful. From these results, we conclude that while interactive AVs may not be universally preferred by students, it is beneficial to offer a variety of learning media to students to accommodate individual learning preferences. We hope that our experiences will be informative for other developers of interactive AVs, and encourage educators to exploit these potentially powerful resources in classrooms and other learning environments.

Keywords: Interactive algorithm visualization; Pedagogy; Design; Evaluation; Human factors; Artificial intelligence

1. Introduction

Artificial Intelligence (AI) is an important discipline within computer science, but it is hard to teach. One major difficulty in teaching AI concepts is that they often involve complex, dynamic algorithms (Hearst, 1994; Greiner and Schaeffer, 2001). Using a blackboard or slides to show algorithm dynamics during lectures, as was done at the University of British Columbia (UBC) prior to introducing CIspace[1] in 1999, was laborious for instructors and ineffective for students.

CIspace is a set of interactive algorithm visualizations (AVs) for demonstrating common AI algorithms. AVs, also called 'algorithm animations' in the literature, are

* Corresponding author. Tel.: +1 778 834 3077; fax: +1 604 822 5485.
E-mail addresses: samershi@cs.washington.edu, samershi@cs.ubc.ca (S. Amershi), carenini@cs.ubc.ca (G. Carenini), conati@cs.ubc.ca (C. Conati), mack@cs.ubc.ca (A.K. Mackworth), poole@cs.ubc.ca (D. Poole).

[1] CIspace: tools for learning computational intelligence. Available at: http://www.cs.ubc.ca/labs/lci/CIspace/.

0953-5438/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.intcom.2007.08.003

tools for animating algorithm dynamics on visual representations. We use the term 'interactive AVs' when emphasizing the difference between AVs that allow learners to actively control the animation and manipulate the visual representation and AVs that only allow for passive viewing of animations. The CIspace project was undertaken with the aim of developing a suite of applets that could be used to make learning AI more effective and enjoyable (Poole and Mackworth, 2001; Amershi et al., 2005). CIspace currently consists of nine Java applets, encompassing many of the topics covered in undergraduate and graduate AI courses, such as search, constraint satisfaction, deduction, planning, machine learning, robot control and belief and decision networks.

Several tools and resources for enhancing AI teaching and learning have been proposed (e.g., at the IJCAI 2001 Workshop on Effective Interactive AI Resources, and at the AAAI 1994 Symposium on Improving Instruction of Introductory AI). While a few resources have been developed (e.g., MIT AI Tools, 2002; AAAI's AI Topics, 2000), the majority of these efforts have now either been abandoned (e.g., Manaris and Russell, 1996; Ingargiola et al., 1994) or have not developed beyond the prototype stage (e.g., Greiner and Schaeffer, 2001). Complicating matters further, the dispersion of existing AI tools across the Web has left instructors (and students) with the problem of searching for appropriate tools for each topic and then learning to use them.

Outside the domain of AI, many AVs exist (see Hundhausen et al., 2002 and Naps et al., 1997 for reviews), originating from a substantial body of research on dynamic visualization in general (see Rieber, 1990 and Tversky et al., 2002 for reviews). We use the term 'dynamic visualizations' (DVs), also called 'animations' in the literature, when referring to this general body of work. DVs encompass algorithm and program visualizations in computer science, as well as visualizations of dynamic processes in the natural sciences and other disciplines (see Rieber, 1990 and Price et al., 1993 for reviews). Despite the abundance of these tools and the belief shared by many educators that AVs can help students learn, widespread adoption of AVs by the academic community has yet to occur (Naps et al., 2002; Rößling and Naps, 2002). Again, the primary obstacles instructors face in adopting AVs include the time to locate, learn and teach students how to use relevant AVs, and to incorporate them in a course (Naps et al., 2003). For students, a major concern is uncertainty about the educational effectiveness of AVs over traditional methods of study (Hundhausen, 2002). 'Effectiveness' in this context refers to not only improved learning, but also increased student engagement, motivation and satisfaction.

In this paper, we aim to illustrate how the CIspace project addresses the aforementioned obstacles faced by instructors and students when teaching and learning AI. Our design follows an iterative process in which we first identify pedagogical and usability goals and then devise and implement techniques to achieve these goals through interactive AVs. Finally, we revise our choices in light of feedback from in-class use, usability evaluations and user studies.

Our rationale for emphasizing the distinction between pedagogical and usability goals is to assist interactive AV designers in determining what features to implement for a specific system. Design features are only effective with respect to a goal, so even if there is conflicting evidence of the effectiveness of a feature, considering the goal that the feature is intended to satisfy can help designers make more informed choices. For example, Rößling and Naps (2002) assert that implementing an incremental rewind feature is important for learning, whereas Saraiya et al. (2004) found that such a feature provided no significant advantages in terms of knowledge acquisition measured by test scores. Although these reports appear to be in conflict, we assert that when the intended goal is to motivate students through active engagement an interactive AV designer should choose to implement such a feature, even if its direct effects on knowledge acquisition are not clear. Existing research often merges goals with the design features that may fulfill these goals, making it difficult for developers to extract the features that are important for the goals of a specific application. For example, Rößling and Naps (2002) list the pedagogical requirements they attempt to meet with their interactive AV system. They suggest that AVs must support built-in interactive prediction features. However, we consider this as a design feature that attempts to meet the more general pedagogical goal of promoting active engagement (Bergin et al., 1996; Naps et al., 2002). This separation can also help to define clear and testable hypotheses, such as whether or not a system or subset of features in a system satisfies a specific goal.

In addition to trying to help guide interactive AV designers, we hope that the results of our evaluations on CIspace presented in this paper will encourage educators to take advantage of CIspace and other interactive AVs in computer science courses. Research on AVs, and DVs in general, have shown mixed results of pedagogical effectiveness (Hundhausen et al., 2002; Naps et al., 2002; Rieber, 1990; Tversky et al., 2002). Most of these have focused on measuring pedagogical effectiveness in terms of knowledge acquisition. Reviews of experiments on DVs (see Hundhausen et al., 2002; Rieber, 1990 for example), have shown that roughly half have reported either non-significant differences between the DVs and the media used in the various control conditions (e.g., static visualizations or text), or significant differences in favor of the control condition. Researchers have offered several hypotheses as to why DVs have failed to produce favorable results. These hypotheses include confounding experimental factors (e.g., excessive difficulty of the lesson content Rieber, 1989), inadequate evaluation methods and measures (e.g., focusing on knowledge acquisition rather than on alternative measures such as motivation, and on quantitative measures rather than on both qualitative and quantitative measures (Hundhausen et al., 2002; Gurka and Citrin, 1996), and individual student differences (e.g., differences in learning style, background knowledge and expertise, spatial ability, and even age (Adams et al., 1996; Stasko et al., 1993; Large et al., 1996; Gurka and Citrin, 1996; Rieber, 1990). Also, some suggest that well-designed static media may simply be just as effective as DVs (Pane et al., 1996; Tversky and Morrison, 2001; Tversky et al., 2002). More optimistic results reported in the literature have shown significant pedagogical advantages of DVs compared to the control conditions. Analysis of these studies reveal that in many of them students engaged in some type of interaction with the DVs, such as manipulating input data or answering questions, while students in the control conditions read text, listened to lectures or passively watched animations (Hundhausen et al., 2002; Rieber, 1990; Tversky et al., 2002). However, Tversky et al. (2002) argue that the lack of equivalence between the experimental conditions (e.g., actively interacting with a DV versus passively reading text) in such experiments negates any conclusions drawn from them about the benefits of DVs. They contend that for two experimental conditions to be comparable they must (1) convey the same information (i.e., the content and level of detail provided by the DV must be the same as that provided by the static visualization or text), and (2) use equivalent procedures (i.e., the level of student engagement with either media should be the same).

In light of these findings, we conducted a range of evaluations on CIspace as is recommended for assessing the pedagogical effectiveness and usability of AVs (Stasko and Hundhausen, 2004). Specifically, we conducted two controlled experiments on CIspace in which we made efforts to design for comparable experimental conditions. The goal of our first controlled experiment was to gauge the effectiveness of interacting with one of our AVs compared with working through sample problems on paper in terms of knowledge acquisition, as this is a traditionally accepted way of measuring effectiveness (Hundhausen et al., 2002). The main conclusion we drew from this experiment was that our interactive AV was at least as effective at increasing student knowledge as the well-established paper-based medium. Our second controlled experiment was inspired by the hypothesis of several researchers that the value of AVs may be made more apparent by using alternative measures of effectiveness, such as preference and motivation (Demetriadis et al., 2003; Hubscher-Younger and Narayanan, 2003; Kehoe et al., 2001). Therefore, the goal of our second controlled experiment was to measure effectiveness in terms of user preference. The main results from this experiment show that students liked using the interactive AV and felt that it helped them learn more than the paper-based medium (these results were statistically significant). However, students were divided when forced to choose a medium to study with. Analysis of comments from questionnaires and semi-structured interviews revealed that certain interface issues with our interactive AV influenced some of the students' choices. These interface issues have since been resolved, or are in our plans for future revisions. Although students were divided when forced to choose a medium, the majority reported that in practice they would use both the interactive AV and the paper-based medium. From these results, we conclude that while interactive AVs may not be universally preferred by students, it is beneficial to offer a variety of learning media to students in order to suit individual learning preferences.

We also present the results of the usability evaluations we conducted on CIspace. A series of semi-formal usability evaluations helped us identify usability problems and guide the design of CIspace during its initial development stages. In addition, we collected data from usability surveys that we distributed to students in two different AI courses at UBC that were using CIspace and that were taught by two different instructors. The main results from these surveys substantiate the claim that students would use the CIspace tools in practice. In addition, students reported that the tools were generally easy to use and useful.

The rest of this paper is organized as follows: Section 2 provides a brief history of DV research, with an emphasis on AV research in computer science education. In Section 3, we discuss the pedagogical and usability goals that we identify as important for CIspace. Section 4 describes the key design features we have included in the latest version of CIspace to help us achieve our goals. In this section, we also introduce constraint satisfaction problems (CSPs) and illustrate some of our design features with examples from the Consistency Based CSP Solver Applet and some of our other applets. In Section 5, we discuss the pedagogical and usability evaluations we have conducted on CIspace. In Section 6, we discuss possible avenues for future research. Section 7 concludes with a summary of the paper.

2. Background

As early as 1966, researchers were experimenting with computer-animated, dynamic visualizations for use as instructional aids in computer science and other disciplines (e.g., Knowlton, 1996). Several cognitive science theories on learning supported the intuition that DVs would be powerful tools for clarifying abstract concepts. For example, Paivio's (1971, 1983) dual coding theory rationalizes the use of visualizations (static or dynamic) as necessary for activating the non-verbal subsystem of the dual brain. According to his theory, visualizations reinforce verbal understandings by enabling the brain's non-verbal or visual subsystem to construct representations of knowledge interconnected with the verbal subsystem. Neglecting either subsystem would be less effective for understanding than activating both simultaneously. Theories on mental models (e.g., Mayer, 1981; Norman, 1983; Johnson-Laird, 1983; West, 1992) also support the use of visualizations to facilitate the development of accurate internal models of abstract concepts and processes (Levy et al., 2003; Byrne et al., 1999). The Epistemic Fidelity Theory asserts that

DVs are ideal for transferring an expert's mental model of a dynamic process to a student (Hundhausen, 1999).

Despite theoretical support for the use of DVs, technological constraints had limited most instructors to textual media and the occasional visual drawn on the blackboard. It was not until the arrival of relatively affordable graphically based personal computers in the late 1970s that DVs for instructional use became feasible. One of the earliest inexpensive systems designed to support teaching and learning in computer science was Dionne and Mackworth's (1978) ANTICS system. ANTICS enabled real-time production of graphically animated LISP programs, which could be controlled interactively by script commands or light pens and function buttons. Yet, it was Ron Baecker's (1981) animated film *Sorting Out Sorting* that is generally recognized for initiating a revival of research in DVs, and particularly AVs, in the field of computer science (Price et al., 1993; Byrne et al., 1999; Baecker, 1998). Equipped with color, sound and narration, this relatively simple animation illustrates the dynamics of nine sorting algorithms on different data structures. In addition, an entertaining race between all nine algorithms at the end of the film allows comparison of computational performance. Although no formal studies were conducted on the film's pedagogical effectiveness, it has been widely used in introductory computer science courses to this day.

Following *Sorting Out Sorting*, there emerged a steady stream of DVs for demonstrating algorithms and programs in computer science (e.g., Brown and Meyrowitz, 1983; Stasko, 1990), for simulating processes in physics and biology, for illustrating algebraic and geometric properties in mathematics, and for instruction in other disciplines (see Rieber, 1990 and Price et al., 1993 for some reviews). A few of these early systems were empirically evaluated for pedagogical effectiveness, yielding a mix of favorable and disappointing results (Rieber, 1990). Soon after, theories on active learning began influencing the design and development of these tools as educators and researchers started recognizing the potential value of making DVs interactive (Brown and Sedgewick, 1984; Cowley et al., 1993; Wilson et al., 1995; Carlson et al., 1996; Rieber, 1990; Hundhausen et al., 2002). Experiential Learning Theory emphasized practice and knowledge application for quality learning (Kolb, 1984); Cognitive Constructivism favored knowledge construction over passive knowledge absorption (Ben-Ari, 1998). Active learning is believed to help motivate and engage (Adams et al., 1996), improve metacognitive learning skills (Naps et al., 1997), and aid in the understanding of the mapping from domain concepts to visualizations (Stasko et al., 1993). One of the first interactive AVs built to support active learning was the BALSA system for Pascal algorithms (Brown and Sedgewick, 1984). Users could start, stop and set the speed of algorithm execution and analyze different views of the algorithm simultaneously. Many other systems also appeared during this time, equipped with innovative means for interaction, including enabling data input and manipulation, encouraging algorithm or process step prediction, and supporting the implementation of custom animations (Rieber, 1990; Price et al., 1993).

Confident in the potential value of interactive DVs, researchers commonly attributed the reasons for these tools not being exploited in classrooms and courses to platform dependency issues and lack of distributive technologies necessary for widespread access (Gurka and Citrin, 1996; Naps et al., 2006). Then, in the second half of the 1990s, with the advent of the Internet, the World Wide Web and Java Virtual Machine (JVM), came the promise of major changes in teaching and learning (Bergin et al., 1996; Boroni et al., 1998, 1999). Educators eagerly anticipated moving from static classrooms to high-tech, interactive and engaging educational environments that relied on DVs to make abstract processes more easily accessible to every student.

Still today, over two decades after *Sorting Out Sorting* made its appearance, and despite ever-increasing technological advances, widespread adoption of interactive DVs by the educational community has yet to occur (Naps et al., 2002; Rößling and Naps, 2002; Hundhausen, 1999). Furthermore, most use of these tools remains limited to passive in-class demonstrations, which is inconsistent with continued belief in the value of interactive DVs (Rieber, 1990; Kehoe et al., 2001; Naps et al., 2002). This is not to say that interest in interactive DVs has stagnated. On the contrary, educators and researchers have continued to make progress in interactive DV technologies, as is evident from the countless DV systems and repositories that have materialized over the years (Rieber, 1990; Ingargiola et al., 1994; Bergin et al., 1996; Naps et al., 1997; Hundhausen et al., 2002). But the lack of an authoritative site to find these tools, along with insufficient attention and reluctance from instructors, have resulted in many of these tools and repositories disappearing.

Obstacles to using interactive DVs in teaching and learning include both pedagogical concerns and usability problems (Tversky et al., 2002; Hundhausen et al., 2002; Ingargiola et al., 1994; Naps et al., 2003). Even with many researchers focusing on developing and evaluating interactive DVs rather than passive DVs, reports on pedagogical effectiveness continue to be mixed (see Hundhausen et al., 2002 or Rieber, 1990 for reviews in computer science and other disciplines). For example, in the literature on interactive AVs in computer science, several researchers have reported that using interactive AVs is just as effective as actively engaging students in learning through other methods. These methods include having them create their own visualizations (Hundhausen and Douglas, 2000), having them role play the execution of algorithms (Rantakokko, 2004), having them predict the behavior of an algorithm using static diagrams (Byrne et al., 1999), and having them learn from well-designed static media (Pane et al., 1996). In contrast, several other researchers have found evidence in favor of interactive AVs. For example, Grissom et al. (2003) showed that student learning increases as the level of student engagement with an AV increases (i.e., interacting with an AV was better than just viewing the AV, which was better than not seeing any AV at all). In another example, a series of eight experiments comparing an interactive AV embedded in a multimedia environment against various control conditions showed that using the AV environment was more effective than using static materials or listening to lectures (Hansen et al., 2002).

While most of the above evaluations focused on measuring effectiveness in terms of knowledge acquisition, several recent studies have looked at other factors that may reveal the benefits of AVs, and DVs in general, including increased levels of student participation and motivation. However, in most of these experiments the methods used to evaluate the DVs in terms of these factors have been either observational or indirect (e.g., measuring time-on-task to show motivation); even then the results have sometimes been mixed (e.g., Pane et al., 1996; Kehoe et al., 2001; Rantakokko, 2004; Hundhausen and Douglas, 2000; Hundhausen, 2002). For example, Kehoe et al. (2001) observed that students using interactive AVs to work through homework-style questions appeared more relaxed and confident than students in a control condition with no animation. They also found that students in the AV condition spent more time on their task than students in the non-AV condition. They argue that these observations indicate increased motivation as a result of using the interactive AV. In contrast, Pane et al. (1996) found no significant differences in student attitudes and preferences for an interactive DV embedded in a multimedia environment over well-designed static materials. They also measured time-on-task and found significant differences in favor of the DV condition; however, they attributed most of this difference to the time required to run the DVs and not to student motivation levels. Conflicting results such as these have made it difficult for educators to justify the use of interactive AVs, and DVs in general, especially when considering the effort needed to integrate them into courses. Usability deficiencies, especially those involving the time required to find appropriate tools and then learn to use them, have also been cited as some of the most common reasons preventing educators from making use of interactive AVs (Naps et al., 2002; Crescenzi et al., 2002). Without instructor support, students fail to benefit from these readily available and potentially powerful tools.

Much work has gone into determining effective interactive AV design features that address some of these pedagogical and usability problems. From this corpus of published research come numerous design recommendations and lists of best practices for interactive AV development (e.g., Naps et al., 2002; Fleischer and Kucera, 2001). For example, Naps et al. (2003) advocate designing AVs specifically for instructor needs, e.g., *capturing larger concepts* to alleviate the time required to search for, install and learn new tools, as well as *developing a supporting Web site* where all relevant instructions and supporting documentation can be assembled. Saraiya et al. (2004) also evaluated and recommended several design features for their pedagogical effectiveness, e.g., example data sets and pseudocode displays. However, because the rationale behind one design feature may sometimes conflict with another, it can be difficult to determine the types of features to implement when creating a new interactive AV. For example, Naps et al. (2003) suggest that an AV should map directly to an existing resource to facilitate course integration. However, they also argue that a more flexible system can ease course integration by being adaptable to a variety of resources and instruction styles.

In light of these issues, we adhere to an iterative approach to development for our interactive AVs, weighing design choices in terms of the pedagogical and usability goals we aim to achieve. We first make our intended goals explicit in order to guide our design. Then, we devise and implement features to achieve these goals. Finally, we revise our choices in light of feedback from in-class use, usability evaluations and controlled experiments. We argue that this scheme is effective, and hope our experiences can inform other developers and encourage interactive AV use.

In the next section, we illustrate the pedagogical and usability goals that form the basis of the CIspace suite.

3. CIspace goals

Our underlying goal in developing CIspace is to enhance traditional approaches to AI instruction. This objective can be broken down into the two broad categories of pedagogical and usability goals. These categories are not completely distinct in that poor usability can mask pedagogical rewards, and limited pedagogical effectiveness can make efforts towards usability irrelevant. Satisfying goals in both categories, however, greatly improves the effectiveness of any educational aid.

Next, we describe the key pedagogical (Section 3.1) and usability (Section 3.2) goals that we aim to achieve in the iterative design of CIspace. Some of these goals are presented in terms of more specific subgoals, expressed as a taxonomy of objectives (see Fig. 1 at the end of Section 3.2).

3.1. Pedagogical goals

For a learning aid to contribute to education, it must provide clear and definite pedagogical benefits. The following are the pedagogical goals informing the design of CIspace.

P1. Increase understanding of the target domain. In the domain of AI this includes understanding the mappings from abstract knowledge to visual representations, as well as understanding the various AI algorithms. Guaranteeing a measurable increase in understanding of domain concepts and processes has traditionally been the focus of many AV researchers (Hundhausen et al., 2002), and we agree that this is a justified goal. However, we maintain that the pedagogical value of an interactive AV is not limited to this specific measure, as is emphasized by the other pedagogical objectives we list in this section.

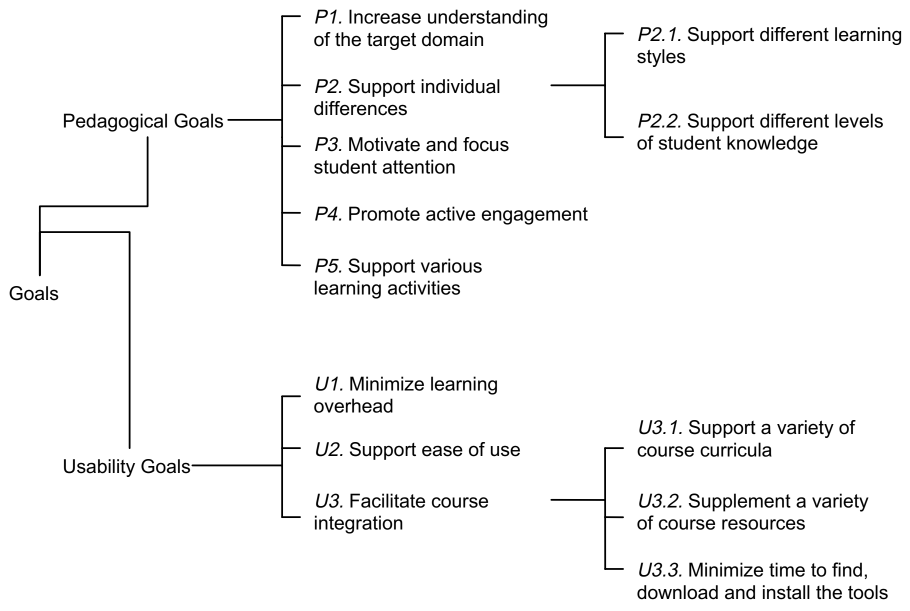

Goals
Pedagogical Goals
P1. Increase understanding of the target domain
P2. Support individual differences
P3. Motivate and focus student attention
P4. Promote active engagement
P5. Support various learning activities
P2.1. Support different learning styles
P2.2. Support different levels of student knowledge
Usability Goals
U1. Minimize learning overhead
U2. Support ease of use
U3. Facilitate course integration
U3.1. Support a variety of course curricula
U3.2. Supplement a variety of course resources
U3.3. Minimize time to find, download and install the tools

Fig. 1. CIspace goal taxonomy.

P2. Support individual differences. Individual Differences Theory (Cooper, 1997; Riding and Rayner, 1998) emphasizes that the learning outcome from a given learning methodology is dependent on distinguishing characteristics of the learner. Characteristics such as learning style, aptitude and background knowledge have been shown to greatly influence the effectiveness of a learning tool (Naps et al., 2003; Adams et al., 1996; Kehoe et al., 2001). Therefore, we adopt the goal of supporting individual differences, which can be divided into the following subgoals:

P2.1. Support different learning styles. Learning theorists have proposed several behavioral models to categorize students by various learning styles (Naps et al., 2003; Adams et al., 1996). For example, Felder's (1993) model identifies four behavioral dimensions of learning: *sensory/intuitive*, *visual/verbal*, *active/reflective*, and *sequential/global*. These inherent learning styles have been found to influence student preferences for different educational media, including AVs, and to shape the learning strategies that students develop for using them (Kehoe et al., 2001; Stern et al., 2005). To accommodate the wide range of students that may comprise a classroom, the design of CIspace should therefore account for such differences in learning style.

P2.2. Support different levels of student knowledge. An individual student's understanding of a subject may vary over time. The rate by which each individual learns can also differ. Bloom and Krathwohl's (1956) well-known taxonomy characterizes individual understanding on six progressive levels:

(1) *Knowledge level.* The student can recall factual information.
(2) *Comprehension level.* The student can comprehend the meaning behind the information.
(3) *Application level.* The student can apply the learned information to new problems.
(4) *Analysis level.* The student can break down a more complex problem and use the learned information to analyze the components.
(5) *Synthesis level.* The student can make generalizations and new inferences from the learned information.
(6) *Evaluation level.* The student can assess the value of the information and make comparisons between competing ideas.

Factors that may contribute to differences in understanding include a student's background knowledge, the difficulty of the subject matter and even language barriers (Adams et al., 1996). To accommodate these diverse levels of expertise, we want CIspace to be able to exercise the skills of both novices and increasingly more advanced students while supporting individual learning pace.

P3. Motivate and focus student attention. Much of the research on AVs has focused primarily on measuring learning gains to demonstrate effectiveness (Hundhausen, 2002), yet results from these studies continue to be mixed. More recently, however, there have been preliminary investigations showing that the value of interactive AVs may lie in their ability to increase student motivation (which may indirectly improve understanding by increasing the time students are willing to spend learning (goal *P*1), improve long-term learning, and alleviate learner stress (e.g., Kehoe et al., 2001 and Demetriadis et al., 2003). We can further argue that motivational factors are necessary to focus the attention of the often-distracted (Grissom et al., 2003; Bergin et al., 1996), technically savvy, MTV and Nintendo generation (Soloway, 1991; Guzdial and Soloway, 2002) of students in today's classrooms. These students may be accustomed to images and other visual media because of their exposure to with technology (Adams et al., 1996). Therefore, interactive AVs that motivate and focus student attention may help satisfy the needs of this generation of students.

P4. Promote active engagement. One way to motivate students (goal *P*3) is by actively involving them in the learning process (Bergin et al., 1996). In the context of interactive AVs, this may be achieved by supporting different forms of interaction between the student and the tool. The ITiCSE working group on 'Improving the Educational Impact of Algorithm Visualizations' (Naps et al., 2002) defined six classes of engagement:

- *No Viewing.* No use of visualizations.
- *Viewing.* Any use of visualizations.
- *Responding.* Using visualizations while answering questions about them.
- *Changing.* Modifying visualizations to explore different instances.
- *Constructing.* Constructing new visualizations.
- *Presenting.* Presenting visualizations for discussion and feedback.

The authors hypothesize that AVs supporting a mix of these activities will produce better learning outcomes for students (goal *P*1). Thus, in designing CIspace we aim to provide features for eliciting many of these forms of engagement while attempting to balance our usability objectives (see Section 3.2).

P5. Support various learning activities. Most educators recognize the benefits of in-class use of AVs (Naps et al., 2002), though the primary role of the student in this scenario is rather passive. In contrast, higher levels of engagement (goal *P*4) with interactive AVs can be attained through activities generally occurring outside of the classroom, such as individual exploration or course assignments (Hundhausen et al., 2002; Kehoe et al., 2001). In these scenarios, students typically become active participants in the learning process by performing activities such as answering questions (e.g., Hansen et al., 2000), exploring different algorithm parameters (e.g., Lawrence et al., 1994), or even constructing new visualizations (e.g., Hubscher-Younger and Narayanan, 2003). Furthermore, using interactive AVs in multiple activities can increase the user's familiarity with the tools, which may make them easier to use and reduce learning time (goals *U*1 and *U*2), and, as Naps et al. (2002) suggest, can result in improved learning from them (goal *P*1). Thus, to take full advantage of interactive AVs, we aim to provide support for various learning activities.

3.2. Usability goals

An educational aid may be designed based on sound pedagogical principles, but without satisfying the usability needs of both educators and students, it would rarely become an effective teaching system. Usability encompasses a number of criteria, including learnability, efficiency and memorability. These are seemingly intuitive objectives, yet usability deficiencies, especially those involving the time to learn and use interactive AVs, are the most cited reasons for educators not adopting these tools (Naps et al., 2003). It is therefore essential to tackle these usability goals in the very early stages of designing a pedagogical system. Here, we describe the usability requirements we have identified as essential for our CIspace tools.

U1. Minimize learning overhead. Ninty percent of educators responding to a survey distributed prior to the ITiCSE 2002 conference cited that the time it takes to learn a new tool is a major impediment to using interactive AVs in a course (Naps et al., 2002). Minimizing learning overhead allows teachers/students to spend less time learning the operations necessary to begin teaching/learning the target domain, and more time actually teaching/learning the target domain. This requires each CIspace tool to be relatively lean, but without compromising our pedagogical goals.

U2. Support ease of use. After learning how to use a tool, it should be easy and efficient for educators and students to carry out their tasks. Davis and Wiedenbeck (2001) studied the effects of the perceived ease of use of software on users. They found that perceived ease of use results in an increase in perceived usefulness, and, for users with some prior exposure to similar interfaces, an improvement in task performance. Therefore, as the primary task of an educational aid is to assist learning, perceived ease of use may help to improve understanding (goal *P*1). Perceived usefulness may also build up instructor confidence in AVs as well as motivate students to use them for learning (goal *P*3).

U3. Facilitate course integration. Educators report in-class demonstrations as the most common use of AVs in computer science courses, with fewer educators incorporating them in homework exercises or making them available for individual exploration (Naps et al., 2002). Problems adapting AVs to individual teaching approaches, course content and other course resources discourage tighter integration of AVs in a course. Thus, while ensuring that a tool is easy to learn (goal *U*1) and use (goal *U*2) can help alleviate instructor effort, ease of course integration is essential so that more educators will be motivated to take full advantage of these potentially powerful resources. This goal can be divided into the following subgoals:

U3.1. Support a variety of course curricula. AI course content can vary across different institutions and amongst individual instructors. For CIspace to be a useful resource for most AI educators, the tools must therefore be flexible enough to suit a variety of AI course curricula. This also means that it should be easy for instructors to create customized AVs for their particular course.

U3.2. Supplement a variety of course resources. We envision CIspace supplementing textbooks or teacher-constructed materials rather than being standalone tools. It is essential, then, that the interactive AVs be compatible with these other resources to effectively reinforce student understanding (Hubscher-Younger and Narayanan, 2003;

Kehoe et al., 2001; Naps et al., 2000). To achieve smooth integration, it must be easy for instructors to create these accompanying materials or combine CIspace with existing resources.

U3.3. Minimize time to find, download and install the tools. Easing course integration also requires that the appropriate tools be easy to find and straightforward to download and install (Naps et al., 2003).

The following taxonomy summarizes the goals described in this section that we aim to achieve in our design of CIspace.

4. CIspace design for pedagogical and usability goals

Since CIspace is an ongoing experiment in pedagogy, we continue to evolve our tools through an iterative approach of design and evaluation. Here, we describe some of the key design features we have incorporated into the latest version of CIspace, and the pedagogical and usability goals they aim to satisfy. These design features are informed by AV research and the pedagogical and usability evaluations we have performed on CIspace to date (see Section 5). While the individual design features we use may not be an original contribution to interactive AV, the comprehensiveness of CIspace is nevertheless unique, particularly regarding interactive AVs for AI. Describing our design choices is important to exemplify our framework for making design decisions based on explicit goals. Table 1 summarizes the mapping between the design features we will describe and the goals discussed in the previous section.

We illustrate the design features with the CIspace Constraint Satisfaction Problem (CSP) applet and, where appropriate, with references to other CIspace applets. We therefore precede the feature descriptions in Section 4.2 with a brief overview of CSPs and the algorithm the CSP applet demonstrates for solving them (Section 4.1). We focus on the CSP applet because CSPs are pervasive in AI, yet simple enough to introduce in a limited space.

4.1. Introduction to CSPs and AC-3[2]

The problem of constraint satisfaction can be stated as follows: given a set of variables each with a domain (a set of values it can take on), and a set of constraints on legal assignments, find an assignment of a value to each variable that satisfies all constraints. The nature of a CSP lends to its intuitive graphical representation as a network of variable nodes and constraint arcs. For example, Fig. 2 shows a CSP designed for scheduling activities A, B, C, D and E at times 1, 2, 3, 4. Vertices represent the activities and their possible domain values, and edges with square boxes represent constraints on activity times.

In research literature, a series of algorithms for solving a CSP by achieving network consistency, known as AC-i, $i = 1, 2, \ldots$, have been proposed. The CIspace CSP applet demonstrates the AC-3 algorithm. Network consistency is reached when all arcs in a network have been made consistent. An arc $\langle X, r\rangle$, where r is a relation $r(X, \boldsymbol{Y})$ on variable X and some tuple of other variables $\boldsymbol{Y}$, is arc consistent if, for each value $x \in \mathrm{dom}(X)$, there is some value $\boldsymbol{y} \in \mathrm{dom}(\boldsymbol{Y})$ such that $r(x, \boldsymbol{y})$ is true. For example, arc $\langle \mathrm{A}, \mathrm{A} = \mathrm{C}\rangle$ in Fig. 2 is consistent because for each domain value in variable A, there is some value in variable C such that $\mathrm{A} = \mathrm{C}$ is true. Arc $\langle \mathrm{B}, \mathrm{B} > \mathrm{C}\rangle$ is not consistent because there exists a value in B that is inconsistent with the relation $\mathrm{B} > \mathrm{C}$ given the available domain values in C; in particular, there is no value in C that is less than 1. The AC-3 algorithm makes the entire network consistent by considering a set of potentially inconsistent arcs initially containing all of the arcs in the network. Until the set is empty, an arc is removed from the set and tested for consistency. If it is found inconsistent, it is made consistent by removing domain values causing the inconsistency, and all consistent arcs that could, as a result, have become inconsistent are placed back into the set. For example, arc $\langle \mathrm{B}, \mathrm{B} > \mathrm{C}\rangle$ can be made consistent by removing 1 from the domain of B.

There are three possible cases that can occur once network consistency has been reached:

- A CSP in which some variable's domain is empty. In this case, the CSP has no solution.
- A CSP in which each variable's domain has a singleton value. Here, the CSP has a unique solution.
- A CSP in which every variable's domain is non-empty and at least one variable's domain has multiple values left. In this case, any non-singleton domain may be split into non-empty sets and then the algorithm can be applied recursively to the resulting sub-problems.

4.2. Design features

Next, we describe key design features of the CIspace applets, referencing the CSP applet and CSP concepts described in Section 4.1, as well as other CIspace applets. We justify our design feature choices in the context of recent work on interactive AV in computer science education.

4.2.1. Accessibility

The CIspace applets are freely available online and are licensed by the University of British Columbia under a Creative Commons license.[3] The licensing allows anyone to use and distribute the tools for non-commercial purposes. Making the tools freely available over the Web offers a number of advantages. First, Web-based tools permit remote accessibility, enabling CIspace to be used in and outside of the classroom which helps to support various

[2] The description of CSPs and the AC-3 algorithm are based on the textbook Computational Intelligence (Poole et al., 1998). For more details, consult this text or almost any other introductory AI textbook.

[3] http://creativecommons.org/licenses/by-nc-sa/1.0/.

Table 1
Mapping of design features to goals

	P1	P2.1	P2.2	P3	P4	P5	U1	U2	U3.1	U3.2	U3.3
Accessibility						√					√
Coverage and modularity									√	√	√
Consistency							√	√			
Graph-based visual representations	√	√		√						√	
Sample problems			√					√			
Create new problems			√		√	√		√	√		
Interaction	√	√	√	√	√	√					
System help							√	√			

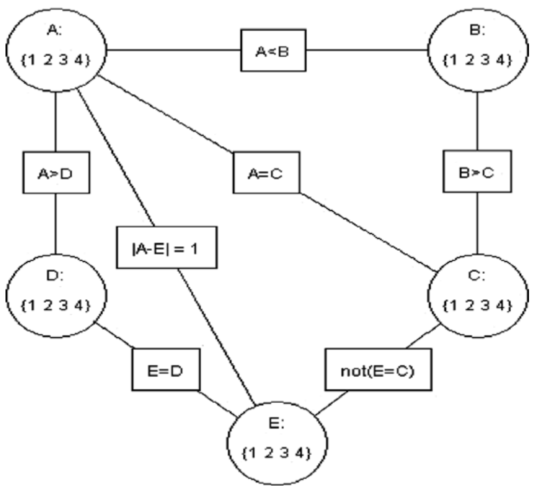


Fig. 2. Example scheduling CSP.

learning activities (goal $P5$). Also, together with the Java Virtual Machine (JVM), Web-accessible Java applets help support platform independence (Naps et al., 1997). Educators cite platform dependency problems as a major impediment to widespread adoption of AVs, second only to issues involving time (Naps et al., 2003). The CIspace applets can be run through most major browsers (provided the JVM has been installed) that support several common platforms including Windows, Mac, Linux and Solaris. Web-based tools that support platform independence can help reach a wide audience of educators and students (Carlson et al., 1996; Rößling and Naps, 2002) in a variety of learning scenarios (goal $P5$). Finally, running Java applets from a Web browser eliminates the need for complicated installations (goal $U3.3$) (Naps et al., 1997, 2003). The tools can also be downloaded as applications for use offline. Here the installation process amounts to downloading and unzipping a CIspace tool and then simply starting the application (goal $U3.3$).

4.2.2. Coverage and modularity

CIspace currently provides coverage of a range of topics traditionally taught in undergraduate and graduate AI courses. Providing coverage helps to overcome the problem instructors and students face in finding AVs for each new topic covered in a course (Naps et al., 2003), and thereby eases course integration (goal $U3$)). While Rößling and Naps (2002) approach this problem by proposing a large general-purpose system that can contain separate AVs for a diverse set of topics, our approach is to provide a modular set of Java applets that each teach a distinct topic and together teach a unified collection of ideas, rather than a large system trying to fulfill (possibly competing) goals. Modularity gives instructors the option to select only those applets that apply to their intended syllabi (goal $U3.1$).

The tools were originally created to complement the textbook *Computational Intelligence* (Poole et al., 1998), and so were modularized based on topics covered therein. However, as each applet encapsulates a unified and distinct set of fundamental AI concepts, CIspace can and has been[4] used to support other popular textbooks, e.g., Russell and Norvig's (2003) *Artificial Intelligence: A Modern Approach*. For instructors, this creates flexibility in choosing other resources (goal $U3.2$).

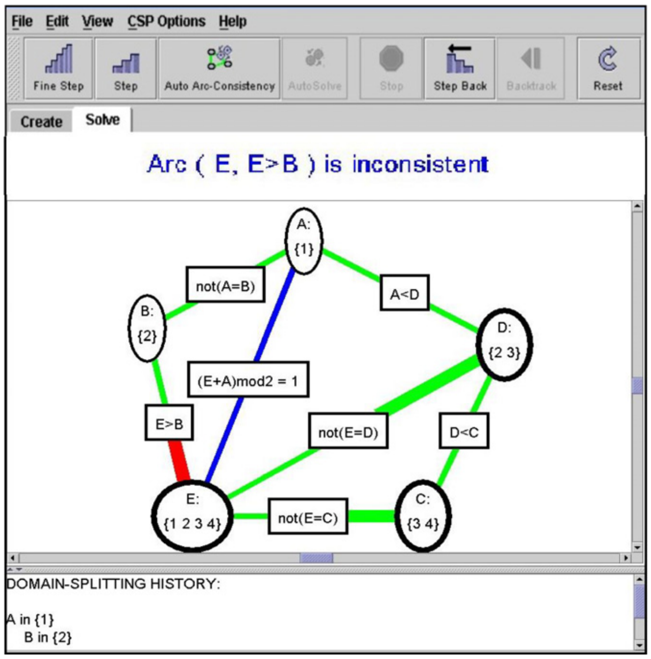
4.2.3. Consistency

A key feature of CIspace is the consistency we attempt to maintain across the applets. The result of this consistency is that users familiar with one applet can transfer experience to other applets, minimizing learning time and facilitating use (goals $U1$ and $U2$). Consistency also reduces the need for instructors to learn possibly highly varied systems authored by different developers in dissimilar styles for each new subject in a course.

Consistency is evident in both the visual representations (see *Graph-Based Visual Representations* design feature below) as well as the interfaces for interacting with these visual representations[5] (see Figs. 3–5). Interface aspects common to all applets include general layout, two separate modes for creating and solving problems, and analogous mechanisms for creating problems and executing algorithms. For example, as with all the CIspace applets, the CSP applet in Fig. 3 is centered on a large canvas where the CSP network is displayed. Above this, a small message panel displays instructional messages about how to use the

[4] See Russell and Norvig Online Demos (Applets) of AI. Available at: http://aima.cs.berkeley.edu/demos.html.

[5] For further interface details, see our Look and Feel document available at: http://www.cs.ubc.ca/labs/lci/CIspace/CIspaceWebDev/CIspace/newlookandfeel/lookandfeel.html.

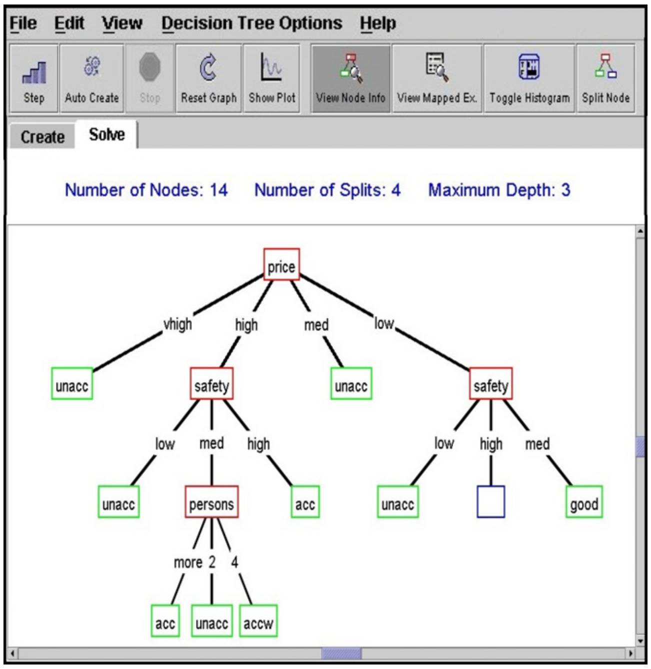


Arc (E, E>B) is inconsistent

DOMAIN-SPLITTING HISTORY:

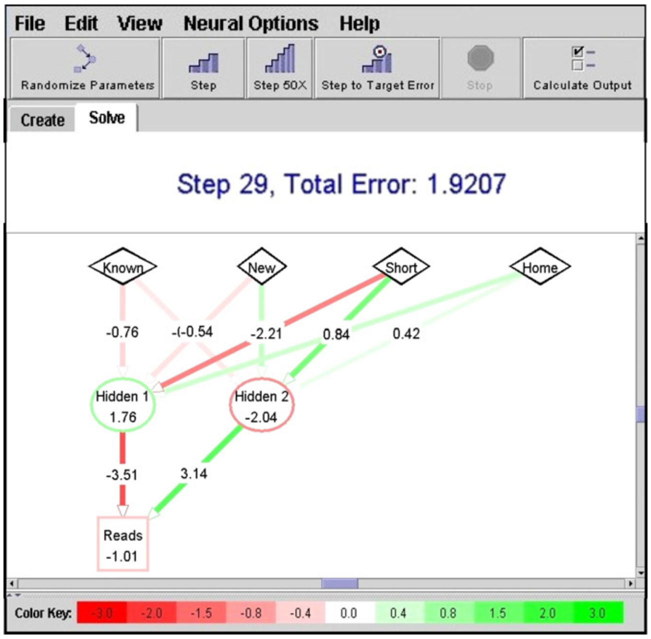
A in {1}
B in {2}

Fig. 3. CSP applet with an example CSP.



Number of Nodes: 14 Number of Splits: 4 Maximum Depth: 3

Fig. 5. Decision tree applet.



Step 29, Total Error: 1.9207

Fig. 4. Neural network applet showing nodes and weighted arcs after learning.

applet or about the current state of the CSP. In *Create* mode, users can build problems using common mechanisms available in the icon based-toolbars near the top of the applet window, and in *Solve* mode users can interactively apply the AC-3 algorithm to the problem.

4.2.4. Graph-based visual representations

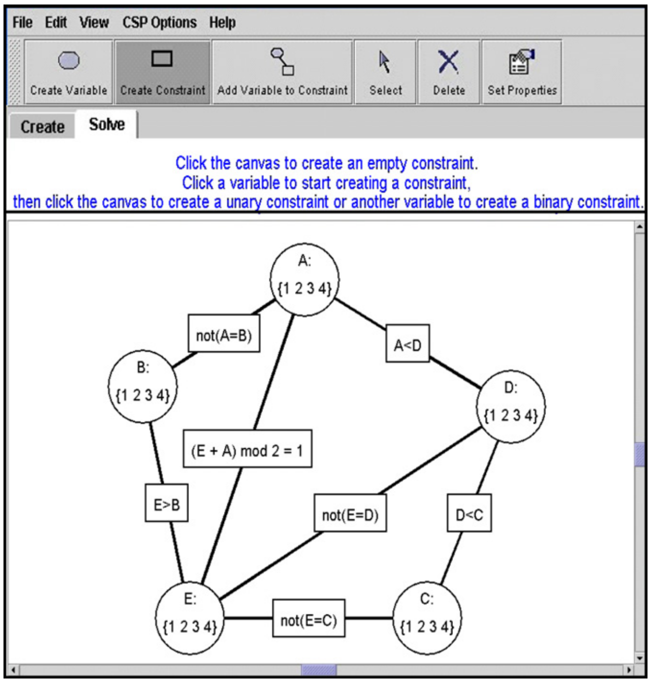
An appropriate graphical representation for each topic forms the foundation of every applet. We chose these representations to be based on the graphical primitives of nodes and edges, as they are adequate for illustrating a wide variety of algorithms. For example, nodes and edges can be arranged as an undirected graph to model a constraint network (see Fig. 3), as a directed acyclic graph (DAG) to model a feed-forward neural network (see Fig. 4), or as a tree to model a decision tree (see Fig. 5). Using these simple and common representations helps ensure that the tools are flexible enough to complement a variety of course resources that may also use classic graphical representations (goal *U*3.2).

The function of these visual representations is to appeal to a wider audience than would text alone (goal *P*3) by helping to make difficult and often abstract concepts concrete (goals *P*1 and *P*2.1). The applets do provide some textual explanations (see the message panels in Figs. 3–5), though they are intended to be used along with text-based materials. Separating the visuals from in-depth textual explanations of theory allows instructors flexibility in choosing other supporting resources and in formulating their own explanations tailored to their individual teaching styles (goal *U*3.2).

4.2.5. Sample problems

Each tool is equipped with a set of sample problems that attempt to highlight salient aspects of a given algorithm or to demonstrate how the algorithm can be used in solving real-world problems. In the CSP applet for example, users can load one of several sample CSPs accessible through the file menu. Each sample problem is designed to illustrate one of the three cases that can occur following arc-consistency, as described in Section 4, or to show how common real-world problems such as scheduling meeting times can be represented as CSPs. In other examples, the Graph Searching applet includes a sample problem depicting part of Vancouver's road network, where the task is to search for paths from the University of British Columbia campus to Stanley Park in downtown Vancouver. The Decision Tree and Neural Network applets both include sample data sets of typical decisions consumers make when buying cars, based on properties such as price, maintenance cost, size and safety ratings. Including sample problems means the tools require little effort to be made usable (goal *U*2). It also reduces the time instructors must spend creating relevant examples (goal *U*2).

The study of a heapsort interactive AV by Saraiya et al. (2004) showed that students who were given sample problems performed significantly better on a post-test than students who had to create their own problems. The authors reason that this is because students generally find it difficult to construct their own problems. However, all of the students participating in the study had limited to no prior knowledge of the algorithm or the relevant data structures. Thus, providing sample problems may be helpful for students new to a subject (goal *P*2.2) (Atkinson et al., 2000) or who find it difficult to construct their own meaningful problems.



Click the canvas to create an empty constraint.
Click a variable to start creating a constraint,
then click the canvas to create a unary constraint or another variable to create a binary constraint.

Fig. 6. CSP applet in *Create* mode.

4.2.6. Create new problems

It is possible that more advanced students could still benefit from creating their own problems. Thus, in order to accommodate the needs of students with different levels of expertise (goal *P*2.2), each applet allows students to experiment with a variety of activities related to problem creation, including inputting new data sets (Neural Network and Decision Tree applets), creating new knowledge bases (Definite Clause Deduction applet), building new environments (Robot Control applet), and constructing new graphs (all CIspace applets). These can be self-directed, as in a study scenario, or instructor-guided through laboratory exercises or assignments. For example, in the latter case students can analyze a given problem, come up with a representation for that problem using the *Create* mode of a CIspace applet, and then explore and answer questions regarding an algorithm's effect on the representation. Such activities can induce a mix of engagement types, including viewing, responding, and changing (goals *P*4).

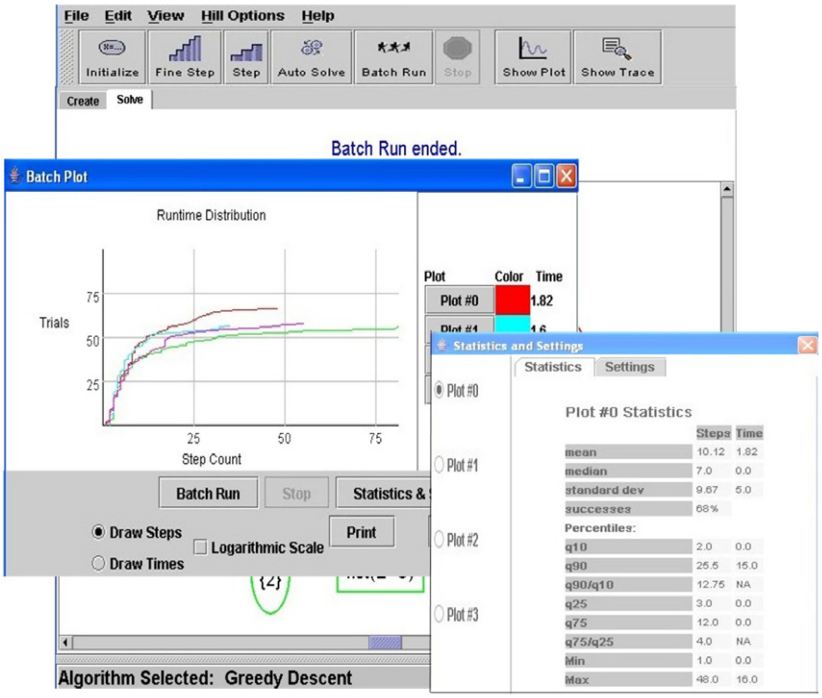
With the CSP applet for example, users can intuitively construct a new CSP through the applet's *Create* mode, which acts like a graphical editor (see Fig. 6). In this mode, users can insert graphical primitives (nodes and edges) onto the canvas to assemble a CSP network by simply selecting the *Create Variable* or *Create Constraint* button, clicking directly onto the canvas, and then specifying variable properties or constraint types as directed by a pop-up dialog. When the user clicks on any button in this mode, the message panel displays instructions on how to proceed with building the CSP. This process is intuitive and useful for building small networks. For larger problems containing many variables and constraints, however, a text editor is available to ease network construction by enabling users to *copy*, *paste* and then edit entity specifications (goal *U*2).

Instructors can use this design feature to create their own problems to show in class or distribute to students (via the Web) for exploration or to use in assignments (goals *U*3.1 and *P*5).

4.2.7. Interaction

While experimental evaluations of AVs have provided mixed results regarding pedagogical effectiveness, most researchers agree that interaction is what is required to increase the pedagogical value of these tools (Rieber, 1990; Tversky et al., 2002; Hundhausen et al., 2002). For example, interaction is what may motivate students, induce active engagement and thus improve learning (goals *P*3, *P*4 and *P*1) (Hundhausen et al., 2002). We believe that interaction also makes the tools appealing for various learning activities both in and outside of the classroom (goal *P*5).

Each CIspace applet provides multi-scaled stepping mechanisms for executing the corresponding algorithms. A user can manually advance through an algorithm at a fine or coarse scale to analyze the visualization state changes at every step. Execution control allows users to learn at their own pace (goal *P*2.2). Users can also run the entire algorithm at once at their preferred speed, or, when non-determinism is involved, execute the algorithm many times in a batch run to see performance statistics (see Fig. 7, right window). In Saraiya et al. (2004), active user control over the execution of an algorithm (goal *P*4) was found to have the most significant pedagogical benefit over other tested design features (goal *P*1).



Algorithm Selected: Greedy Descent

Plot	Color	Time
Plot #0	🟥	1.82
Plot #1	🟦	1.6

Plot #0 Statistics

	Steps	Time
mean	10.12	1.82
median	7.0	0.0
standard dev	9.67	5.0
successes	68%	
Percentiles:		
q10	2.0	0.0
q90	25.5	15.0
q90/q10	12.75	NA
q25	3.0	0.0
q75	12.0	0.0
q75/q25	4.0	NA
Min	1.0	0.0
Max	48.0	16.0

Fig. 7. Stochastic local search applet's batch plot and performance statistics windows.

For illustration, we now explain in detail the mechanisms provided by the CSP applet for executing the AC-3 algorithm on a CSP network. Unless otherwise stated, each mechanism is activated by clicking on its corresponding button on the button toolbar (see top of Fig. 3) or menu item accessible through a pop-up menu that appears by right clicking on the applet's large white canvas. Note that in the CSP applet, network arcs that need to be tested for consistency are colored blue,[6] inconsistent arcs are colored red, and consistent arcs are colored green. The mechanisms for executing the AC-3 algorithm include the following:

- *Fine Step:* Allows users to apply and analyze detailed steps of the AC-3 algorithm. *Fine Stepping* has three stages carried out by three consecutive clicks of the *Fine Step* button or pop-up menu item. Initially, all arcs in the network are blue and need to be tested for consistency. In the first stage, the applet automatically selects a candidate blue arc, which then appears highlighted in the network. In the second stage, the applet tests the arc for consistency. If it is found to be consistent, the arc's color will change to green and the *Fine Step* cycle terminates. If it is inconsistent, its color changes to red and a third *Fine Step* is needed. In this final stage, the applet reduces the domain of the connected variable to remove the inconsistency and turns the arc green. Arcs that could have become inconsistent as a result of this domain reduction need to be retested and are again turned blue. The effect of each *Fine Step* is reinforced explicitly in text through the message panel display (see text above graph in Fig. 3).
- *Step:* Executes the algorithm in coarser detail. One *Step* performs all three stages of *Fine Step* at once.
- *Direct Arc Click:* Clicking directly on an arc in the network activates the *Direct Arc Click* mechanism, which is equivalent to a *Step* on that arc. This mechanism gives users control over the algorithm by allowing them to choose which arcs to make consistent rather than having the applet select arcs for them, as happens with the *Fine Step* and *Step* mechanisms.
- *Domain Split:* Clicking directly on a variable, or network node, brings up a dialog box listing all of the domain values available for that variable. Within the dialog box, the user can specify which domain values to keep and which to set aside. This reduces the CSP to a sub-problem that can then be solved. The applet keeps track of which sub-problem is being solved by recording all domain splits in the *Domain Splitting History* panel at the bottom of the applet (see Fig. 3).
- *Backtrack:* Recovers the alternate sub-problem set aside by *Domain Splitting* and updates the *Domain Splitting History* to reflect the current sub-problem.
- *Auto Arc-Consistency:* Automatically *Fine Steps* through the CSP network until it is consistent. The user can specify the pause duration between successive *Fine Steps* through the *CSP Options* menu. A faster speed is useful in giving the user an overall picture of the AC-3 algorithm; a slower speed enables users to better observe details of the algorithm. As described in Section 4.1, once a network is made consistent the user may still need to split domains to find a solution.

[6] For interpretation of the references to color in this figure, the reader is referred to the web version of this paper.

- *Auto-Solve:* Iterates between making the CSP consistent (by *Fine Stepping*) and automatically splitting domains until a solution is found. If the CSP has more than one solution, then activating the *Auto-Solve* mechanism again will first *Backtrack* to the sub-problem that was set aside during the last automatic *Domain Split*, and then iterate again between making the CSP consistent and domain splitting until another solution is found, and so on. A record of the *Domain Splitting*, *Backtracking* and solutions found is displayed in the *Domain Splitting History* panel for reference.
- *Back Step:* Steps backwards through the AC-3 algorithm. Each *Back Step* reverses the effects of one *Step*.

The granularity of algorithm execution determines the degree of abstraction that a user can examine. Felder's (1993) model of learning behaviors identifies *sequential* vs. *global* learners: sequential learners prefer to first understand the details of an algorithm and then progress linearly towards understanding its overall effects, whereas global learners prefer to initially abstract away details and learn in larger jumps, or understand the big picture and overall goal first. Such behavioral dichotomy was observed by Stern et al. (2005), in a study where students using an interactive AV for review proceeded in either a top–down or high-level manner or in a bottom–up manner. To cater to users with different preferences (goal *P*2.1), CIspace provides multiple levels of abstraction.

4.2.8. System help

All of the CIspace applets include several levels of help designed to address the objectives of *U*1 and *U*2. Each applet provides guidance for carrying out tasks, in the form of carefully placed messages suggesting how to proceed at any given point during the interaction. Each applet is also accompanied by a set of increasingly detailed help pages:

- *QuickStart:* Contains only the necessary information needed to start using the applet quickly.
- *General Help:* A reference page explaining every applet feature and mechanism.
- *Tutorials:* Step-by-step instructions detailing how to complete specific tasks.

Also, in our pedagogical experiments (see Section 5.1), we developed a 3-min instructional video that received positive feedback from the study participants. On average, these participants reported spending less than ten min learning to use the applet being evaluated, including watching this video. This led us to develop video tutorials for all of the applets to complement the text-based tutorials. These videos range from three to seven min in duration and include narrated screen captures of specific tasks being performed.

A summary of the mapping between CIspace objectives and design features is provided in Table 1 at the beginning of this section. As the table shows, each goal is achieved by at least two design features. We argue that this level of redundancy provides an adequate foundation for a robust and reliable set of tools.

5. Evaluation

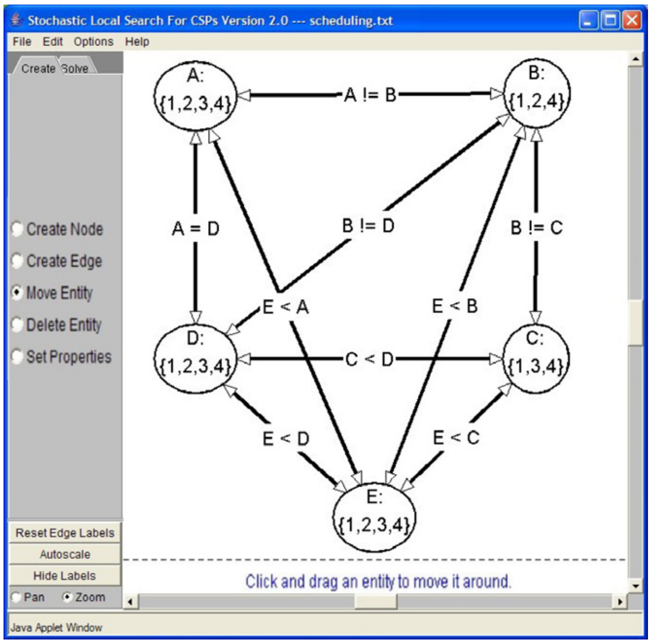
The mapping between goals and design features described in the previous section (see Table 1) was informed by intuition, research, and the evaluations performed on CIspace to date. Since the introduction of CIspace in 1999, the tools have been deployed in over 20 undergraduate and graduate level AI courses at the University of British Columbia (UBC) and used by over 750 computer science students. The tools have been used for in-class demonstrations and assignments as well as for general study. Thus far, CIspace has been well received by both the course instructors and students at UBC. We also continue to receive positive feedback from educators and students internationally. Though the response to CIspace has been encouraging, formal evaluations are necessary to provide robust evidence of pedagogical effectiveness and usability.

In the following sections, we present the pedagogical and usability evaluations[7] we have performed on CIspace in the sequence that they were conducted. First, in the summers of 2001 and 2003 we conducted a series of semi-formal usability tests on each of the CIspace applets (see Section 5.1). The purpose of these tests was to identify usability issues and to refine the design of the applets. Next, in the summer of 2004 a controlled experiment was conducted on the CIspace CSP applet (discussed in Section 4.1) to determine its pedagogical effectiveness in terms of knowledge acquisition (goal *P*1) (see Section 5.2). Then, in the winter of 2005, we collected data from students in an advanced undergraduate AI course at UBC using the Belief and Decision Network applet (Section 5.3). The purpose of this evaluation was to assess the usability of the tools in a natural setting. In particular, we wanted to determine how students typically learn to use the tools in a course (goal *U*1) and to assess their ease of use (goal *U*2). In the spring of 2005, we performed a second controlled experiment on the CSP applet, this time measuring learning preference (goals *P*2.1) (see Section 5.4). Finally, in the summer of 2005 we conducted a second in-class evaluation where we collected usability data from students using the CSP applet in an introductory undergraduate AI course at UBC (Section 5.5). Eventually, we would like to evaluate the degree to which we have achieved each of the goals described in Section 3.

5.1. Evaluation 1: Semi-formal usability testing

To help guide the design of CIspace and identify usability issues, we conducted a series of semi-formal usability

[7] Raw data from all evaluations can be found at http://www.cs.ubc.ca/labs/lci/CIspace/CIspacePapers/rawdata/RawExperimentalData.htm.



Click and drag an entity to move it around.

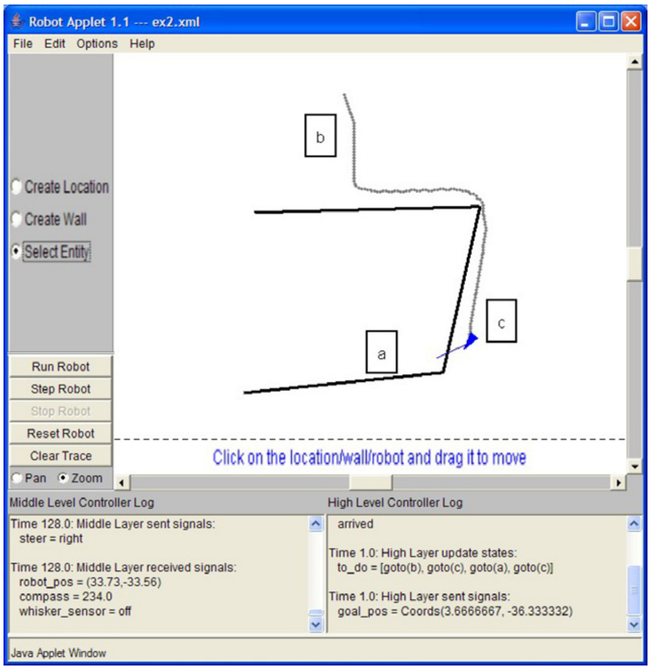
Fig. 8. Original stochastic local search applet.

tests on the applets as we were developing them. In the summer of 2001, four of the applets (the Belief and Decision Network applet, CSP applet, Graph Searching applet, and Stochastic Local Search applet) were tested by at least two and as many as seven volunteer undergraduate and graduate students from the UBC Computer Science Department. Some of the participants were novice users of the applets, while others had prior experience using them. Each participant was given a brief written introduction to the AI topic of the applet that they were testing. They were also given a short description of how to use the applet and were told that they could find additional information about it from the *Help* menu. After reading the introductory information, each participant was given a list of typical tasks to perform using the applet. For example, the Stochastic Local Search (SLS) applet (Fig. 8 shows the original version of the SLS applet used during the usability tests) was designed to demonstrate several different search algorithms for solving constraint satisfaction problems. Typical tasks students were asked to perform included selecting a particular search algorithm and setting specific algorithm parameters, loading a sample problem, *Fine Stepping* and *Stepping* to learn how the algorithm selects variables and chooses domain values, comparing the performance of two different search algorithms by executing batch runs on the problem, and creating a new constraint satisfaction problem from scratch. The participants were given unlimited time to read the introductory material and perform the tasks. An experimenter observed each participant and recorded noticeable bugs, usability problems and participant comments.

The applet-specific bugs that surfaced during these tests were subsequently resolved. Also, some general usability issues were noted. For example, participants were not noticing the messages in the message panel, which was originally located below the graph in the applet window (see Fig. 8), even though several tactics were used to draw attention to the panel whenever a new message was displayed (e.g., the panel would flash yellow and the color of the message text would change). As a consequence, many participants were not aware of some of the applet mechanisms, even though messages about them were displayed in the message panel. For example, some participants testing the CSP applet did not realize that they could simply click on an arc in the network to make it consistent, even though the following message was displayed in the message panel: “Click on an arc to make it arc consistent.” Also, some participants were confused about how to use certain applet mechanisms. For example, participants were confused about the *Domain Splitting* and *Backtracking* mechanisms of the CSP applet (see Section 4). After *Domain Splitting*, students are supposed to continue making the network arc consistent to solve the problem. However, two participants were observed clicking the *Backtrack* mechanism after *Domain Splitting* as if it would solve the problem. Another usability issue that arose was that some participants found that the applet buttons and radio buttons were not intuitive. For example, some participants had trouble trying to move graph entities around. They stated that the “Move Entity” radio button was not as clear as a familiar select icon (i.e., an arrow) would be.

To address these usability issues, we carried out a system-wide redesign of all of the applets in 2003. The message panel was moved above the graph to make it more noticeable, and additional messages were added to help guide students on how to use the applet mechanisms that they had found confusing (e.g., after a user splits a variable’s domain, the following message appears in the message panel: “Click another variable to split its domain or press *Fine Step* or *Step* to continue”). Also, the original buttons were replaced with icon-based buttons with text labels to make their functionality more clear.

In addition to addressing these usability issues during the redesign, we found several inconsistencies between applets at this time. First, not all of the applets had two distinct ‘Create’ and ‘Solve’ modes. This inconsistency can be observed by comparing the original version of the SLS applet in Fig. 8 with the original Robot Control applet in Fig. 9. In the original Robot Control applet, there was no separation between the mechanisms for creating a problem (e.g., see the “Create Location” and the “Create Wall” mechanisms in Fig. 9) and solving a problem (e.g., see the “Run Robot” and the “Step Robot” mechanisms in Fig. 9). Second, some of the mechanism names were inconsistent across the applets. For example, the “Move Entity” mechanism in the SLS applet (see Fig. 8) and the “Select Entity” mechanism in the Robot applet (see Fig. 9) were functionally equivalent but had different names. These inconsistencies would have made it difficult for users familiar with one applet to attempt using a different applet. Therefore, to facilitate our goals of minimizing learning



```
Time 128.0: Middle Layer sent signals:
  steer = right

Time 128.0: Middle Layer received signals:
  robot_pos = (33.73,-33.56)
  compass = 234.0
  whisker_sensor = off
```

```
  arrived

Time 1.0: High Layer update states:
  to_do = [goto(b), goto(c), goto(a), goto(c)]

Time 1.0: High Layer sent signals:
  goal_pos = Coords(3.6666667, -36.333332)
```

Fig. 9. Original robot control applet.

overhead (goal *U*1) and supporting ease of use (goal *U*2), we adhered to stricter consistency standards when redesigning the applets (see Section 4.2 – *Consistency*). These standards are defined in the CIspace look-and-feel document that can be found on the CIspace Website at: www.cs.ubc.ca/labs/lci/CIspace/CIspaceWebDev/CIspace/newlookandfeel/lookandfeel.html.

After redesigning the applets, we did another round of semi-formal usability tests in the summer of 2003. We tested all of the applets with at least two volunteer graduate students in the Computer Science Department at UBC, following the same procedure as in the first round of tests in 2001. All of the participants had experience using the original version of the applets. During the tests, some of the participants commented that they liked the consistency across the new applets since it made them easier to use. Participants also noticed and read the messages in the message panel in its new location above the graph area.

Some new general usability issues also surfaced during these tests. First, most participants did not notice information placed at the bottom of the applet window. For example, participants did not notice the information in the *Domain-Splitting History* panel at the bottom of the CSP applet (see Fig. 3) until it was pointed out. This is still an issue present in some of the applets, which we plan to address in future revisions of CIspace. It should be noted that as a temporary solution during our controlled experiments (discussed in the following sections), the information at the bottom of the applet windows was explicitly pointed out in a video tutorial watched by the participants before using the applet so they would not overlook this important information. We also observed several difficulties participants experienced using the Planning applet. When creating a planning problem, users must specify an initial state and a goal state. Currently, users specify each state in the Planning applet's *Create* mode by switching between two submodes; many participants found this confusing. We considered having three separate modes for the Planning applet (i.e., a "Create Initial State", a "Create Goal State" and a "Solve" mode), but this would have violated our *Consistency* design feature. Therefore, we have temporarily placed the Planning applet in a beta section of the CIspace Website until this problem can be addressed.

5.2. Evaluation 2: Controlled experiment measuring knowledge acquisition

One function of the CIspace tools is to help students learn AI concepts by example, since studying by example is a conventional method of learning (Atkinson et al., 2000; van Lehn, 1998). Therefore, the primary goal of our first controlled experiment in the summer of 2004 was to determine the pedagogical effectiveness of one of the CIspace applets, the CSP applet with arc consistency (see Section 4.1), in terms of knowledge acquisition when compared to a more traditional method of studying sample problems on paper. A secondary goal of this experiment was to determine the time required by students to learn and use the applet.

The experiment typified a study scenario in which students learn underlying theory and application from a textbook, study related examples, and finally, are tested for understanding of both conceptual and procedural knowledge. We therefore assessed the pedagogical value of our interactive AV in much the same way as performance in a course is traditionally evaluated (Hundhausen et al., 2002).

The experiment was a between-subject study, with the means of studying the sample problems as the independent variable. The two conditions for the independent variable were sample problems studied using the applet and written sample problems studied on paper, referred to as the applet and non-applet group, respectively. The static, paper-based sample problems used by the non-applet group were carefully designed by experienced AI instructors based on the traditional methods of teaching algorithm dynamics used before the introduction of CIspace in 1999. These problems displayed fine-grained, discrete algorithm steps equivalent to those demonstrated by the AV. In the applet group, students could interactively control the algorithm's execution, while in the non-applet group students were allowed to manually work through the algorithm (i.e., by writing on the paper). These experimental conditions were as informationally and procedurally equivalent (Tversky et al., 2002) as we could make them while still allowing comparison of interactive AV to non-AV media.

5.2.1. Materials

It should be noted that prior to the experiment, we conducted pilot studies to reveal potential problems with our study design or materials. Logging data from the applet and participant comments during these pilot studies showed that students were confused about some of the applet mechanisms even though they were all described and demonstrated in a 3-min instructional video provided to the students. Therefore, given the short amount of time the students had to learn to use the applet and then learn the material, we decided to remove some of the applet mechanisms for this experiment and for our second controlled experiment (Section 4) so that the students would not be overwhelmed. The mechanisms we removed were *Step* and *AutoSolve*. The reduced version of the applet still included multi-scaled stepping mechanisms for interactive learning and still supported both *sequential* and *global* learning styles (see the *Interaction* design feature in Section 4.2), since we retained mechanisms such as *Fine Step* (aimed at sequential learners) and *Auto Arc-Consistency* (aimed at global learners). Note also that the *Back Step* mechanism (described in Section 4.2) was not implemented at the time our controlled experiments (both this experiment and our second experiment, described in Section 5.4) were conducted.

All of the study participants were given photocopied text about CSPs from the textbook *Computational Intelligence* (Poole et al., 1998). They were provided with two sheets of blank paper on which they could write notes if they wished. In order to guide their study, participants were also given a list of topics to try to learn.

Participants were given either the CSP applet to use to study three sample problems, or the same sample problems written on paper. Each sample problem illustrated one of the three cases that can occur once arc-consistency is achieved, as described in Section 4.1. The written sample problems were modeled after the way CSP examples were illustrated in AI courses by two experienced professors at UBC prior to the introduction of the CIspace applets (see Appendix A). The applet group was also given a 3-min video describing how to use the applet and the applet's interface, but not providing extra information about CSPs.

The pre and post-tests used in the study were comparable (see Appendix B). The tests contained both procedural questions (e.g., "Make the given network arc consistent and give all solutions") and conceptual questions (e.g., "Explain why domain splitting is useful to solve this problem"). The maximum mark for both tests was 19, with 10 marks for the procedural-type questions and 9 marks for the conceptual-type questions.

We also administered condition-specific questionnaires (see Appendix C), in which students were asked about the following:

- Their confidence in their knowledge of the topics given to them at the start of the study. A 5-point Likert scale (where 5 represented *Excellent* and 1 represented *Poor*) was used to rate each topic.
- Their opinions about the study materials used and how those materials influenced their learning of the subject, also using a 5-point Likert scale (5 = *Agree*, 4 = *Somewhat Agree*, 3 = *Neutral*, 2 = *Somewhat Disagree*, 1 = *Disagree*).
- Timing-related questions formatted in ranges of time, e.g., *more than enough, enough, barely enough, not enough* or *less than 5 min, 5–10 min, 10–20 min, over 20 min*.
- Open-ended interface-specific questions (applet group only).

5.2.2. Procedure

A total of 19 students (8 female and 11 male) were recruited for this experiment. Participants were all undergraduate students at UBC who had never taken an AI course but had the prerequisites needed to enroll in UBC's introductory AI course, including a course on basic algorithms and data structures. The experiment took 3 h and participants were paid $10/h for their time.

All of the students were initially given the textbook material, the list of study topics and sheets of paper for taking notes. One hour was allotted to read and study the text. The students were then given 20 min to take the closed-book pre-test.

After the pre-test, the students were randomly divided into the two groups, accounting for balanced distribution of males and females. The applet group had 10 people (6 males and 4 females) and the non-applet group had 9 people (5 males and 4 females). All of the students were given 40 min to study the three sample problems. They could do so in any order and could go back and forth between them. The students were also given back their text material and notes from the earlier learning phase, which they could refer to while studying. The students in the non-applet group were allowed to work through their sample problems by writing on their paper-based materials. During the applet group's study time, the students watched the instructional video, having been told that they could watch it as many times as they liked.

Next, each group was given 20 min to take the closed-book post-test. Finally, the groups were given their respective questionnaires to fill out, with no time limit.

5.2.3. Discussion of results

The pre-test and post-test scores of the applet and non-applet groups showed that both groups improved significantly (Student's t-test, $p < .015$ and .005, respectively) after studying the sample problems, but that there was no statistically significant[8] difference in improvement between the two groups. For the conceptual questions, the non-applet group improved 3% more than the applet group, but the difference was not significant. For the proce-

[8] For all statistical tests, significance is measured at a p-level of .05. Marginal significance is measured at a p-level between .05 and .1.

dural questions, both groups improved by the same amount (33%). The average confidence levels reported by both groups on the list of topics covered were roughly equivalent for each topic, with no significant differences observed. These results show that students are able to learn as effectively with the applet as with studying using paper-based sample problems (goal *P*1), and that they can successfully transfer their knowledge gained from using the applet to a traditional pencil and paper test. This is an important finding because it demonstrates that instructors can incorporate interactive AVs into the studying portion of their courses and still test students in traditionally accepted ways (goal *U*3).

Table 2 shows the results of questions from the questionnaire about students' opinions on the study materials they used. The groups generally agreed that their respective form of studying sample problems helped them to learn the material from the book. We argue that this is a very encouraging result, showing that the perceived effectiveness of CIspace is as good as that of traditional paper-based means developed over the years by the highly experienced AI instructors on our team. The only significant difference between groups (Student's *t*-test, $p < .04$) was in response to the question asking students about the alternate format of study to the one that they used. The applet group's response were between *Somewhat Disagree* and *Disagree* when asked whether they believed that having the sample problems written down on paper would have helped them learn better than with the applet. The non-applet group, on the other hand, was *Neutral* when asked whether they believed watching the CSP graph change at every step would have helped them learn better than with the written problems. The non-applet group was not shown the applet.

On average, both the applet and non-applet groups reported having between *more than enough time* and *enough time* to study their sample problems (goal *U*2), with no significant difference being found. The applet group reported taking between *less than 5 min* to between 5 and 10 min to learn the applet's interface. In fact, all of the students in the applet group reported that it took them under 10 min to learn the interface, with enough time remaining to effectively study the sample problems within the allotted time period (goal *U*1). This finding shows that it takes students little time to learn to use our interactive AVs, contrary to a common reservation that students may be discouraged from using interactive AVs because of the apparent learning overhead involved.

The main results from this experiment are as follows:

- The applet is at least as good as well-established methods of studying examples on paper, given equal study time.
- Students were able to transfer knowledge gained using the applet to a traditional paper test format.
- All students reported taking under 10 min to learn the applet's interface, including watching the 3-min instructional video, and still had *enough* to *more than enough* time to study the sample problems within the given time limit.

5.3. Evaluation 3: Usability survey in advanced AI course

While laboratory studies are useful for controlled testing, the practical value of any educational tool should be evaluated in a natural setting. In the winter of 2005, we collected data from 29 students enrolled in an advanced undergraduate AI course at UBC who were using the CIspace tools. The students were completing an assignment on belief networks, and were given the option of using the Belief and Decision Network applet, without being required to use it. The students were asked to fill out a usability questionnaire about the applet if they used it, or to simply state that they did not use it. All of the students used the applet for the assignment. When asked how they learned to use the applet, 96.6% (28/29) of students reported that they learned by exploration, while 93.1% (27/29) of students reported that they also learned by watching the in-class demonstration on using the applet.

We also asked students to rate the applet (on a 5-point scale) in terms of how easy it was to use, how useful it was, and how enjoyable it was. Table 3 shows the average student ratings for each of these attributes. On average, students found the applet easy to use and useful. They also felt using it was reasonably enjoyable.

We found positive correlations (measured by Pearson's correlation coefficient) between the student ratings for these attributes. We observed that ease of use was strongly correlated with usefulness ($r = .302$, $p < .055$) and enjoyability ($r = .392$, $p < .017$). Usefulness was also strongly correlated with enjoyability ($r = .488$, $p < .0037$). These results are consistent with those found by Davis and Wiedenbeck (2001). Improving ease of use (goal U2), therefore, may increase perceived usefulness, and in turn improve learning performance (goal P1). These results further suggest that ease of use and perceived usefulness affect enjoyability, which in turn may increase motivation for learning (goal P4).

Table 2
Student responses, 5-point Likert scale (5 = Agree, 4 = Somewhat agree, 3 = Neutral, 2 = Somewhat disagree, 1 = Disagree)

Statement	Applet group	Non-applet group
The applet/paper sample problems helped me learn the material from the book.	4.90	4.89
It was difficult to follow steps of the algorithm with the applet/paper sample problems.	2.00	2.44
Looking at examples worked out on paper would have helped me study better.	1.80	N/A
Seeing the network change at every step would have helped me study better	N/A	3.00

Table 3
Average ratings for the Belief and Decision Network applet in terms of ease of use, usefulness and enjoyability, 5-point Likert scale (5 = Best rating, 1 = Worst rating)

Question	Average rating/5
Ease of Use?	4.17
Usefulness?	4.29
Enjoyability?	3.71

We found no significant correlations between students' marks on assignments and these attributes.

5.4. Evaluation 4: Controlled experiment measuring preference

In our first pedagogical evaluation (Evaluation 2, Section 5.2) of CIspace, we designed an experiment comparing knowledge acquisition through studying with the CSP applet against studying sample problems on paper. We found that the applet was just as effective for learning as the traditional method of studying sample problems on paper. However, the advantage of interactive AVs may lie in their ability to engage and motivate students to learn. Few formal studies on AVs have addressed motivation and preference, or have done so only through indirect measures (e.g., Hansen et al., 2002) or through general observations (e.g., Greiner and Schaeffer, 2001; Kehoe and Stasko, 1996). For example, Hansen et al. (2002) suggest that the increased time students spent using an AV compared to text-based study materials is an indication of motivation. This result, however, only indirectly supports user preference for interactive AVs. Observations by Kehoe et al. (2001) showed that a group of students using interactive AVs seemed more relaxed and open to learning than a control group. After the study, the control group students were shown the AVs and then asked to comment on them. Many responded that they believed the AVs would have more positively affected their learning, but these opinions were made retrospectively and without the students having been exposed to the AVs in an actual study setting.

Our second controlled experiment in the summer of 2005 was a within-subject study designed to directly measure user preference for studying with the CSP applet or with sample problems on paper. For this experiment, we augmented the traditional within-subject experiment by first exposing students to both study conditions, and then allowing them to explicitly choose one of the two to study further. We thus obtained explicit quantitative preference data to complement more traditional preference self-reports. Again, the control condition (studying with the sample problems on paper) was designed to be as informationally and procedurally equivalent (Tversky et al., 2002) to the AV media as we could make it (see Section 5.2).

5.4.1. Materials

The materials used for this experiment were the same as those used in Evaluation 2 (see Section 5.2), except that we modified the questionnaire to produce a more in-depth assessment of user preferences and motivation. The questionnaire was divided into two smaller questionnaires so that the students would not be overwhelmed (see Appendix D). The first questionnaire focused on attitudes of students, including assessments of:

- How they liked both forms of study (5-point Likert scale: 5 = *Agree*, 4 = *Somewhat Agree*, 3 = *Neutral*, 2 = *Somewhat Disagree*, 1 = *Disagree*);
- Their perceived learning using both forms of study (5-point Likert scale);
- Their motivation during both treatments (5-point Likert scale);
- Their attitudes towards both forms of study described by semantic differential scales (i.e., ranges between opposing adjectives) including *confusing/clear, boring/exciting, pleasing/annoying,* and *unhelpful/helpful*;
- The amount of effort they felt they put into the study (7-point Likert scale, where *1*represented *none at all* and *7* represented *a great deal*); and
- What materials they would use to study for a test on CSPs given all the materials from the study (i.e., the CSP applet, the paper sample problems, and the text).

The second questionnaire was similar to that used in Evaluation 2, which included questions about applet interface issues, clarity of written sample problems and time taken to learn the applet and study. Also, a brief semi-structured interview was added at the end of the experiment to obtain richer data. The interviews were designed to explore why students chose a particular form of study and how they typically study for tests.

5.4.2. Procedure

A total of 32 students (25 male and 7 female) participated in this experiment. We required participants to be computer science or engineering students and to have taken at least one second year computer science course. We felt that this requirement provided students with sufficient background to learn about CSPs.

The experimental procedure was the same as in Evaluation 2, except for the phase when students studied sample problems. In this phase, all students studied two sample problems for 12 min each, using the applet for one problem and the paper form for the other. A group of 18 students started with the applet, while 14 started with the paper form to account for ordering effects.[9] The 12 min included

[9] The imbalance between the two experimental conditions is a result of running several sessions of the user study, with all the participants in a particular session either starting with the applet or with the paper format, in order to facilitate the study process for the experimenter. Time constraints prevented us from running additional sessions to correct the imbalance (i.e., to run another session with the students starting with the paper format).

Table 4
Average responses, 5-point Likert scale (5 = Agree, 4 = Somewhat Agree, 3 = Neutral, 2 = Somewhat Disagree, 1 = Disagree)

Statement	Average
I liked using the applet more than studying with the sample problems on paper	3.66
I liked studying with the sample problems on paper more than with the applet	2.84
Using the applet helped me more than the sample problems on paper	3.66
The sample problems on paper helped me study better than the applet	2.91

the time for the applet users to view the 3-min video, as in Evaluation 2. Students were then told that there was a third sample problem to study, and were given the choice to study it using either the applet or the paper format. Students were allocated 16 min to study the third sample problem, using their medium of choice. This problem was allotted more time than the previous problems because it illustrated the most complex case of domain splitting with the AC-3 algorithm (Section 4.1).

Following the experiment, in addition to answering the questionnaires, all students were individually interviewed by the same experimenter, with the interview recorded on tape.

5.4.3. Discussion of results

Table 4 shows important results obtained from this experiment pertaining to student attitudes towards the study materials. Overall, the students indicated that they liked using the applet more than the sample problems on paper and that it was more helpful. Both of these statements received significantly more indications of agreement than the opposing statements (paired Student's *t*-test over opposing statements, $p < .032$ and $p < .031$, respectively) and the magnitude of the difference was medium[10] in both cases (Cohen's $d = .587$ and .62, respectively).

The applet was chosen for studying the third sample problem over the paper medium by 19 out of 32 students (12 of the 18 students who started with the applet, and 7 of the 14 who started with the paper sample problems). While more students did choose the applet, this result was not statistically significant and therefore appears to contradict the results on user preference reported above. Analysis of student comments in the questionnaires and the semi-structured interviews provides some explanation for this discrepancy. Four students who chose the paper format to study the third sample problem gave a higher rating for the applet when asked either which format they liked more, or which format they felt helped them to learn more. Some of these students commented that one of the reasons they chose the paper format is because the applet did not have a mechanism for stepping backwards through the algorithm (recall that this mechanism was not implemented in the CSP applet at the time we conducted the controlled experiments). The most common comment made by the students who chose the paper format to study is illustrated by the following dialog:

> *Interviewer:* "I noticed you chose to use the paper to study the last sample problem. Why would you say you chose this format?"
> *Participant:* "I think I myself am not an applet learner. I would rather use the paper because I'm slow and I usually have to go back a few times and the program doesn't really allow me to do that".

A *Back Step* mechanism has since been implemented and is available in the current version of the CSP applet described in Section 4.

The second most common comment made by the students who chose the paper format was that they did so because it allowed them to take notes, whereas the applet does not include a built-in feature for taking notes. One of the four students who chose the paper but gave higher ratings for the applet commented that "The paper would encourage me to make notes on paper and go through each step in more detail." We discuss the possible addition of a note-taking or annotating feature to the CIspace applet in the section on future work (see Section 6).

Finally, an issue concerning the 3-min instructional video may explain why one of the four students chose the paper but gave higher ratings for the applet. This student commented that "The software was helpful, but the Auto Arc-Consistency button was too fast and it doesn't have a stop/pause button." This is interesting considering that the CSP applet does have speed controls as well as a *Stop* mechanism, both of which were demonstrated in the 3-min instructional video provided to participants. This comment may be explained by this student's response to the following interview question:

> *Interviewer:* "Did you find the applet hard to figure out how to use?"
> *Participant:* "A little bit. I needed some help. The video was good, but not good enough".

Therefore, this student may have benefited from additional help features when using the applet. Some possibilities include an intelligent tutoring feature, which we discuss in the section on future work (see Section 6).

While students were divided when forced to choose between the applet and the sample problems on paper, most of them indicated that in practice they would use the applet to study with. In one of the questionnaires, we asked the participants how they would go about studying for a test on CSPs given all the materials they were presented with during the study: the textbook, the applet, and the sample problems on paper. Students were allowed to select as many formats as they wanted for this question.

[10] Cohen's standard suggests that $d = .2$, .5 and .8 are small, medium, and large effects, respectively.

Table 5
Materials students would use to study with in practice (applet choice)

Participant	CSP applet	Paper sample problems	Textbook
1	√		
2	√	√	
3	√	√	√
4	√		√
5	√		
6	√	√	√
7	√	√	√
8	√	√	
9	√		
10	√		√
11	√		
12	√	√	√
13	√	√	√
14	√	√	
15			√
16	√	√	√
17	√	√	√
18	√	√	√
Total (Number/%)	17/94.4	11/61.1	11/61.1

Table 6
Materials students would use to study with in practice (paper choice)

Participant	CSP applet	Paper sample problems	Textbook
19			√
20	√	√	√
21	√	√	√
22		√	√
23	√	√	√
24	√	√	√
25	√	√	
26	√	√	
27		√	√
28	√	√	√
29		√	√
30	√	√	√
31	√	√	√
Total (Number/%)	9/69.2	12/92.3	11/84.6

Of the 31 participants, 83.8% (26/31)[11] said they would use the applet (Sign test,[12] $p < .0001$) suggesting that students would be motivated to study with the applet in practice (goal *P*4); 74.2% (23/31) said they would use the sample problems on paper to study (Sign test, $p < .01$); and 70.9% (22/31) said they would use the textbook (Sign test, $p < .029$), showing that in practice traditional materials would still play an important role even, with the availability of the applet. A closer analysis of these responses (see Tables 5 and 6) reveals that of the students who chose to use the applet for the last sample problem, 94.4% (17/18) said they would use it to study for an actual exam (Sign test, $p < .0001$) compared with 61.1% (11/18) who stated they would use the paper sample problems and 61.1% (11/18) who would use the text. Of the students who chose to use the paper medium, the responses were slightly less divergent but still showed indications of preferred learning media: 92.3% (12/13) stated that in practice they would use the sample problems on paper (Sign test, $p < .003$), 84.6% (11/13) stating they would use the textbook (Sign test, $p < .02$), and 69.2% (9/13) stated they would use the applet.

We argue that these results on user preferences should encourage use of interactive AVs in an effort to support a variety of learners (goal *P*2.1). Rather than restricting possibly more visual or active learners to text-based or static materials, offering students a choice seems to be the most pedagogically beneficial option.

Tables 5 and 6 show the different combinations of materials students stated they would use to study for an exam in practice. Of the students who chose the applet to study the third sample problem (Table 5), only 22.2% (4/18) said they would use it alone; 72.2% (13/18) said they would use it in conjunction with the paper sample problems, the text, or both. Only 1 student stated that they would use the paper sample problems alone, while no student said they would use the text alone or in combination only with the paper sample problems. For these students, therefore, interactive AVs represent a clear improvement over traditional media alone. Of the students who chose the paper medium (Table 6), 69.2% (9/13) stated they would use the applet in conjunction with the text-based materials, while 30.8% (4/13) said they would use either the textbook alone, or the textbook plus paper sample problems. This result suggests that the majority of students who chose the paper medium may still be motivated to use the applet if it was available, which is also evident in comments from some of these students:

> *Participant:* "[The applet] certainly is a lot of fun. I would use it before I read the text."
>
> *(Different) Participant:* "It would have been more helpful to have the paper version together with the applet".

Tables 7 and 8 show the most interesting results from the semantic differential scales in Questionnaire 1. In these tables, the rows are divided between students who

Table 7
Attributes describing the CSP applet

	8 = Clear, 1 = Confusing	8 = Exciting, 1 = Boring
Applet choice	7.06	5.89
Paper choice	5.69	4.46
Overall averages	6.48	5.29

[11] One student did not complete the questionnaire, so the results presented here include only the responses of the 31 students who did complete the questionnaire. The student who did not complete the questionnaire chose to use the applet to study the third sample problem. Therefore, the results for the group that chose the applet are reported out of 18, though 19 students actually chose it.

[12] The Sign test is a binomial test used to determine if we can reject the null hypothesis that the probability of either of two observations occurring is equally likely. Significance is measured at a *p*-level of .05, while marginal significance is measured between .05 and .1.

Table 8
Attributes describing the written sample problems

	8 = Clear, 1 = Confusing	8 = Exciting, 1 = Boring
Applet choice	5.22	3.28
Paper choice	6.39	4.23
Overall averages	5.71	3.67

chose to use the applet and students who chose to use the sample problems written on paper. These results show that, on average, the students found the applet more clear and exciting than the paper sample problems (the difference was statistically significant, given a paired Student's *t*-test on student ratings for each media, $p < .039$ and $p < .0001$, respectively). The magnitude of the effect size of the applet ratings compared to the ratings for the paper medium was small on the *clear/confusing* scale (applet mean = 6.48 and standard deviation = 1.52, paper mean = 5.71 and standard deviation 1.865, Cohen's $d = .456$) and was large on the *exciting/boring* scale (applet mean = 5.29 and standard deviation = 1.65, paper mean = 3.67 and standard deviation = 2.02, Cohen's $d = .872$). Clarity is an indication that the students find the applet easy to understand and use (goal *U*2), and it is reasonable to presume that an exciting tool may better motivate students to learn than a more boring tool (goal *P*4).

The students' opinions on the amount of time needed to study the sample problems agreed with the first controlled experiment (see Section 5.2). Most students felt that they had between *enough* and *more than enough* time to study each sample problem, and, on average, took between *less than 5 min* and *5–10 min* to learn how to use the applet (goal *U*1).

Finally, both the group of students who chose the applet and the group of students who chose the paper format showed significant improvements in scores from pre-test to post-test, with no significant differences between the groups. In this experiment, we also asked students to rate the level of effort that they felt they put into the study. According to Hundhausen et al. (2002), the more effort required by a learning tool, the better the learning outcome. We found that the group that chose the paper rated their effort level higher than the group that chose the applet. This difference is marginally statistically significant (Student's *t*-test $p < .06$), and the magnitude of the effect of the paper format on student effort compared to that of the applet is medium sized (applet choice group mean = 5.72 and standard deviation = 1.12, paper choice group mean = 6.31 and standard deviation = .85, Cohen's $d = .583$). Given this result, it is surprising that the students who chose the paper-based method did not improve more from pre to post-test than the students who chose the applet. However, because this was a within-subject study, it is difficult to tease out the factors that may have lead to this difference. Furthermore, it is unclear whether the students interpreted this question as asking how much effort they put in independently or as a result of the materials they used. That is, the effort level could reflect a student's proclivity to study, or may indicate that the student found a certain medium required more effort to learn from. In the latter case, the trend could then be attributed to the paper format being more confusing or less exciting to use, as indicated in the results discussed earlier.

The main results from this experiment can be summarized as follows:

- On average, students liked studying with the applet and felt that it helped them learn more than the paper medium.
- Students were divided when choosing a medium to study with.
- The majority of students would use the applet to study with if it were available.
- On average, students found the applet more clear and exciting than the paper medium.
- Students took less than 10 min to learn to use the applet, including watching the video, and still had enough time to study.
- Students who chose the paper medium to study with felt they put in more effort during the experiment than those who chose the applet.

5.5. Evaluation 5: Usability survey in introductory AI course

In the summer of 2005, we again collected usability data from students in an actual course. This time the students were in an introductory undergraduate AI course at UBC that used CIspace and was taught by a different instructor. The students in this course were completing an assignment on CSPs and were required to use the CIspace CSP applet (see Section 4.1). The students were asked to fill out a voluntary usability questionnaire concerning the applet. Eleven students turned in this questionnaire. As in our first in-class evaluation with the advanced AI course (Section 5.3), students were asked how they learned to use the applet. Only one student in the introductory course stated that he learned to use the applet by looking at the CIspace *Help* pages, while no students in the advanced AI course reported learning this way. In contrast with the advanced AI course, where 93.1% of the students reported that one of the ways they learned to use the applet was by watching the in-class demonstration, here only two students (18.2%) reported that they learned by watching the in-class demonstration. This difference may be a consequence of the instructor in the introductory course spending less time using the applet in-class than in the advanced course. Alternatively, students may simply prefer learning by exploration, considering that 10 students (90.9%) reported that they learned by exploring the applet on their own, which is consistent with the majority of students (96.6%) in the advanced AI course learning to use the applets on their

Table 9
Average ratings for the CSP Network applet in terms of *ease of use*, usefulness and enjoyability, 5-point Likert scale (5 = Best rating, 1 = Worst rating)

Question	Average rating/5
Ease of Use?	3.77
Usefulness?	4.45
Enjoyability?	3.64

own. This result may also indicate that the textual messages in the message panel were enough to guide the students in completing their assignments.

Since learning by exploration appears to be the most common means by which students learn to use the applets in practice, our focus on improving the learnability and ease of use of the applets should be on adding more tips and hints in the message panel or on creating additional interface features to help guide the students. For example, some of the students in the introductory AI class commented that the speed of the *Auto Arc-consistency* and *Auto-Solve* mechanisms was too slow. These students were likely unaware of the speed controls available under the *CSP Options* menu. A similar incident occurred during our second controlled experiment (see Section 5.4), when one of the students who chose to use the paper-based sample problems reported that the applet was helpful but that the *Auto Arc-Consistency* mechanism was too fast. Additional messages in the message panel could help direct students to the available speed controls in the applet. Alternatively, an intelligent help feature that could recognize the student's intention and provide advice about the speed options could also help in this regard. We discuss such an intelligent help feature in the following section on future work.

We also found similar results concerning the *ease of use*, *usefulness* and *enjoyability* of the CSP applet (see Table 9) as we found in the advanced course, where students used the Belief and Decision Network applet (see Section 5.3). Also, positive correlations between *ease of use*, *usefulness*, and *enjoyability* were again found. *Ease of use* was strongly correlated with *usefulness* ($r = .608$, $p < .024$) and *enjoyability* ($r = .596$, $p < .027$), and *usefulness* was also strongly correlated with *enjoyability* ($r = .696$, $p < .008$).

6. Future work

We are currently assessing the effectiveness of CIspace in supporting various learning styles (goal *P*2.1). By collecting learning-style data from students taking an introductory AI course at UBC, we are able to examine how style affects performance on assignments involving the CIspace tools. In the future, we intend to conduct further experiments evaluating CIspace in terms of each of our pedagogical and usability goals.

We continue to iterate through our design process and improve CIspace based on results from our evaluations, and on advances in both technology and pedagogical research. For example, recently in the field of software visualization there has been some interest in alternate representations to classic node-link diagrams such as the graphical networks used in the CIspace applets. Ghoniem et al. (2004) compared node-link diagrams with matrix-based representations of graphs, where the rows and the columns of the matrices are indexed by graph nodes and the cells of the matrices are non-zero if a link exists between the corresponding nodes. Additionally, the value of the cell can express a property of the node, e.g., a cost. In their experiment, they showed that for small graphs, node-link diagrams are always more readable and more familiar than matrices. They also found that independent of graph size, node-link diagrams are always better than matrices in supporting the user in finding paths in a graph (and arguably in any tasks involving paths). Matrix-based representations appear to become more effective only on large graphs (i.e., graphs with greater than 50 nodes). These findings support our use of common node-link diagrams for teaching graph-based algorithms in CIspace at least at the beginning when the graph size is typically small and when identifying and processing graph paths (e.g., search). However, in the future, and as larger problems are considered, it may be worth investigating matrix-based or alternative representations for CIspace.

Our evaluations of CIspace revealed that some students were unaware of existing applet features (e.g., speed controls for *Auto Arc-consistency* and the *Stop* mechanism) even when they were pointed out prior to the applets being used. We also found that the most common method students learn to use the CIspace applets is by exploration. Therefore, it may be useful to include an intelligent tutoring feature within each applet that could recognize potential problems and explicitly point students towards useful applet mechanisms while they are exploring. In addition to helping students use the applets, such a feature could also provide students with personalized support for learning the AI algorithms. Since the CIspace tools are unstructured, open learning environments, students using the tools for independent study require metacognitive abilities (Brown, 1987), such as planning of learning activities, self-explaining algorithm behaviors and self-monitoring progress, to learn effectively. Traditional learning is often scaffolded by instructors who prompt students to explain, answer questions or monitor student progress when necessary. An intelligent tutoring feature within CIspace could provide this additional scaffolding to those students who may be inexperienced or have less proficient metacognitive abilities. Therefore, in future iterations we plan to add the subgoal *Support different metacognitive abilities* to pedagogical goal *P*2 (*Support individual differences*). We may also develop an intelligent tutoring feature for each applet to achieve this goal and to better achieve the existing goals of supporting learnability (goal *U*1) and ease of use (goal *U*2). We have taken a first step towards an intelligent tutoring feature for CIspace in (Amershi and Conati, 2006), where we used a machine learning approach to build an on-line classifier that detects student interaction patterns that are detrimental to learning.

Another way that we could help *support different metacognitive abilities* is by including a quiz feature in all of the applets. Some of the CIspace applets already have a quiz feature, which allows users to test their knowledge about the current algorithm. This feature included in all applets could help students monitor their progress (a metacognitive skill) as well as stimulate active engagement (goal *P*4). We plan to include such a feature in all of the applets in future revisions of CIspace.

One of the most common reasons participants in our second controlled experiment (see Section 5.4) gave for choosing the paper-based medium over the applet was that the paper allowed them to easily take notes. Plaisant et al., 1999) designed an environment that facilitated user annotation on records of learner activity kept by the system (called 'learning histories'). They suggest that such a feature could help students learn better by allowing them to review and correct mistakes in their actions, as well as monitor their progress. Krebs et al., 2005) designed a similar AV environment with annotation capabilities, to provide an easy way for instructors to give students feedback about their learning. In future revisions, we may include similar annotation capabilities, so that students and instructors can easily take notes or give feedback about the AVs.

In addition, we are currently pursuing two promising areas of development to better achieve some of our existing pedagogical and usability goals. First, we envision developing user-customizable applets whose interfaces can be tailored. Each applet would include a menu listing its available mechanisms. When given the option, the user (typically the student) would be able to select which mechanisms to keep. The interface would then change according to the user's selections. To guide users in selecting mechanisms that may be helpful for learning given their level of domain knowledge, we could provide default settings for beginner, intermediate and expert users (goal *P*2.2). This would essentially create layered interfaces (Schneiderman, 2003) for the CIspace tools so that users are not overwhelmed by the large number of options when they start using the system (goals *U*1 and *U*2).

Second, we are developing author-customizable applets for authors creating content for a course, book, tutorial or other Web-based document. These customizable applets can be distributed as stand-alone tools or embedded in a Web document inline with text and hypertext. To facilitate the creation of these custom applets, we are developing Web-based forms where authors can simply select the applet mechanisms and interface style aspects they wish to include, and the form will then automatically generate the appropriate html code needed to call the customized applet in an authored document. For instructors developing their own resources, this feature is intended to further our goals of creating tools that are easy to use and integrate into a course (goals *U*2 and *U*3). For students, this feature could be used to create reports and present visualizations for discussion, a highly active form of engagement (goal *P*4) suggested by Naps et al. (2002). Furthermore, enabling the interactive AVs to be used together with textual explanations or other forms of media may, according to Paivio's (1971, 1983) Dual-coding Theory, increase the pedagogical value of the AVs (goal *P*1). This approach may also cater to a wider range of learning preferences and styles, as some students may feel more comfortable learning with textual explanations than with interactive AVs alone (goal *P*2.1).

7. Conclusions

In this paper, we have discussed our design and evaluation of the CIspace interactive AVs for teaching and learning fundamental Artificial Intelligence algorithms. Our design approach iterates by identifying pedagogical and usability goals, introducing design features to achieve these goals, and then revising our choices in light of evaluations. We have compiled a taxonomy of pedagogical and usability objectives to help guide the design of CIspace. These goals aim to address some of the primary educational concerns and usability deficiencies cited by instructors and reported in the AV literature. We have also described and illustrated the key design features that we implemented for the CIspace tools based on this taxonomy. We advocate differentiating between design goals and design features to help designers make more informed choices when developing interactive AVs. We have also detailed the most interesting findings from the pedagogical and usability evaluations we have conducted on CIspace to date. Finally, we have discussed possible avenues for future work on CIspace. We hope that our efforts and results will help inform developers of future interactive AVs and encourage instructors to exploit them in courses in order to enhance teaching and learning.

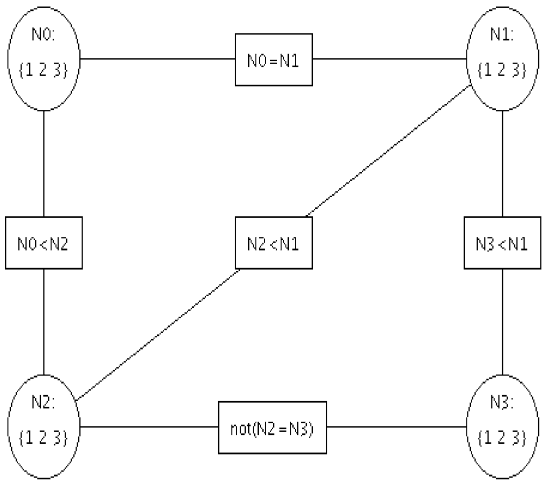
Acknowledgements

The CIspace project has been supported by the Natural Sciences and Engineering Research Council of Canada (NSERC). We thank former members of the CIspace team, including Nicole Arksey, Mike Cline, Wesley Coelho, Peter Gorniak, Holger Hoos, Heather Maclaren, Kevin O'Neill, Mike Pavlin, Kyle Porter, Joseph Roy Santos, Shinjiro Sueda, Leslie Tung, Audrey Yap and Regan Yuen.

Appendix A. Written sample constraint satisfaction problems

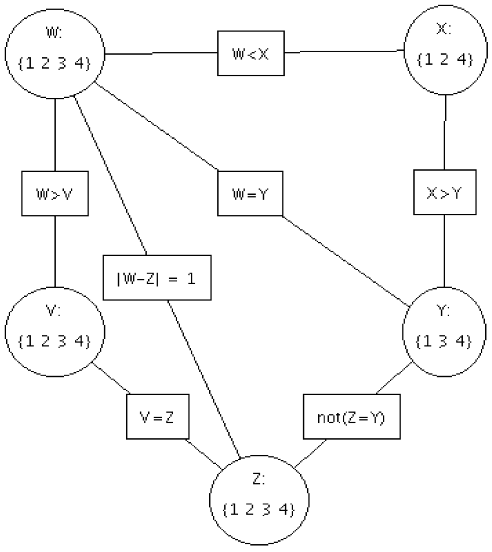
- Below are three sample CSPs for you to study.
- For each sample problem, there is a graph of the CSP and a table showing the steps of the AC-3 algorithm. Each line of the table indicates which arc is currently being considered and which domain values, if any, are removed by the algorithm. For example, in line 3 the arc "(N2, N0 < N2)" is being considered, where the first term, N2, is the variable and the second term, N0 < N2, is the constraint. The domain value 1 is removed from N2 in this step of the AC-3 algorithm. If no domain values are removed, a '-' will be shown under the Element Removed column for that step.
- In CSPs with domain splitting, a graph is shown to display the state of the CSP after backtracking. In addition, the variable domain split is indicated under the Domain Split column.

Sample 1:



	Arc	Element Removed
1	(N1, N0=N1)	-
2	(N0, N0 = N1)	-
3	(N2, N0 < N2)	N2 = 1
4	(N0, N0 < N2)	N0 = 3
5	(N1, N0 = N1)	N1 = 3
6	(N1, N3 <N1)	N1 = 1
7	(N0, N0 = N1)	N0 = 1
8	(N2, N0 < N2)	N2 = 2
9	(N3, N3 <N1)	N3 = 2,3
10	(N3, not(N2=N3))	-
11	(N2, not(N2=N3))	-
12	(N1, N2 < N1)	N1 = 2
13	(N1, N0 = N1)	N0 = 2
14	(N2, N0 < N2)	N2 = 3
15	(N3, N3 <N1)	N3 = 1
16	(N3, not(N2=N3))	-
17	(N2, not(N2=N3))	-
18	(N1, N2 < N1)	-
19	(N2, N2 < N1)	-
	No Solution	

Sample 2:



	Arc	Element Removed
1	(X, W < X)	X = 1
2	(W, W < X)	W = 4
3	(V, W > V)	V = 3,4
4	(W, W > V)	W = 1
5	(X, W < X)	X = 2
6	(Z, V = Z)	Z = 3,4
7	(V, V = Z)	-
8	(Y, not(Z =Y))	-
9	(Z, not(Z=Y))	-
10	(Y, X > Y)	Y = 4
11	(Z, not(Z=Y))	-
12	(X, X >Y)	-
13	(Y, W =Y)	Y = 1
14	(Z, not(Z=Y))	-
15	(X, X >Y)	-
16	(W, W =Y)	W = 2
17	(X, W < X)	-
18	(V, W > V)	-
19	(Z, |W-Z| = 1)	Z = 1
20	(V, V = Z)	V = 1
21	(W, W > V)	-
22	(Y, not(Z =Y))	-
23	(W, |W-Z| = 1)	-
	Solution found: V =2, W=3, X=4, Y=3, Z =2	

Sample 3:

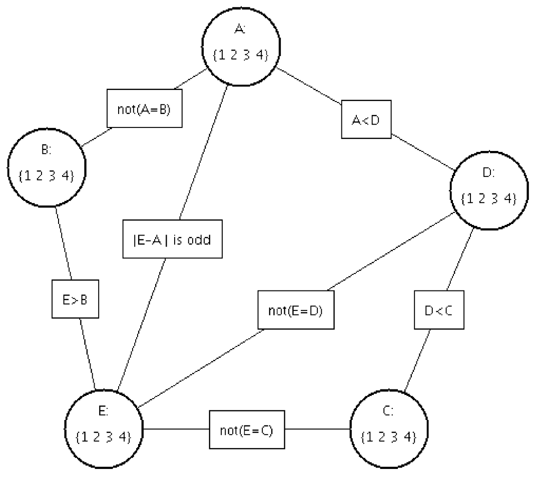


Figure A: Lines 1-33

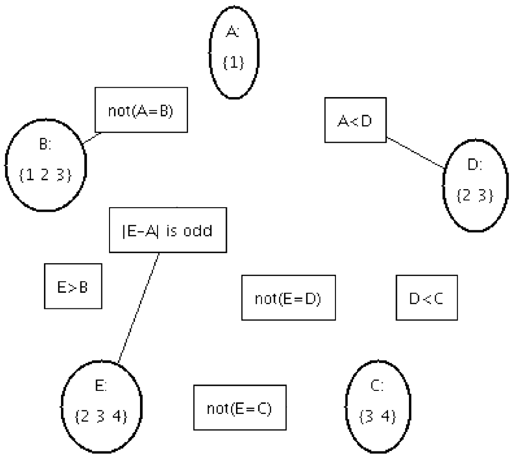


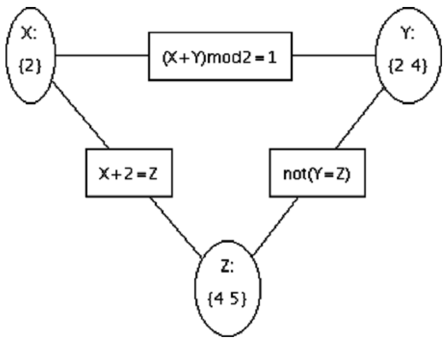
Figure B: Lines 34-35

	Arc	Element Removed	Domain Split/ Backtrack
1	(C, D < C)	C = 1	
2	(D, D < C)	D = 4	
3	(D, A < D)	D = 1	
4	(C, D < C)	C = 2	
5	(A, A < D)	A = 3,4	
6	(B, not(A=B))	-	
7	(A, not(A=B))	-	
8	(D, not(E=D))	-	
9	(E, not(E=D))	-	
10	(C, not(E=C))	-	
11	(E, not(E=C))	-	
12	(B, E > B)	B = 4	
13	(A, not(A=B))	-	
14	(E, E > B)	E = 1	
15	(D, not(E=D))	-	
16	(C, not(E=C))	-	
17	(A, |E-A| is odd)	-	
18	(E, |E-A| is odd)	-	
			A in {1}
19	(D, A < D)	-	
20	(B, not(A=B))	B = 1	
21	(E, E > B)	E = 2	
22	(D, not(E=D))	-	
23	(C, not(E=C))	-	
24	(A, |E-A| is odd)	-	
25	(E, |E-A| is odd)	E = 3	
26	(D, not(E=D))	-	
27	(C, not(E=C))	C = 4	
28	(D, D < C)	D = 3	
29	(A, A < D)	-	
30	(E, not(E=D))	-	
31	(B, E > B)	-	
			B in {2}
32	(A, not(A=B))	-	
33	(E, E > B)	-	
	Solution found: A =1, B=2, C=3, D=2, E =4		
			B in {3}
34	(A, not(A=B))		
35	(E, E > B)		
	Solution found: A =1, B=3, C=3, D=2, E =4		

Appendix B. Tests

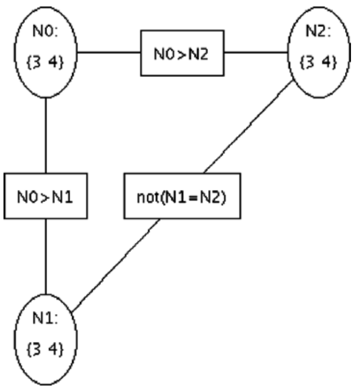
B.1. Pre-test

(1) Consider the following constraint network. Note that $(X+Y) \bmod 2 = 1$ means that $X+Y$ is odd.



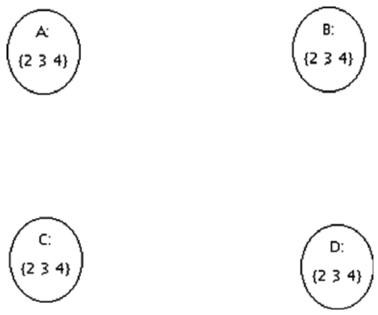
(a) Is this constraint network arc consistent?
(b) If it is, explain why the constraint network is arc consistent. If it isn't, state which arcs are not arc consistent and explain why the constraint network is not arc consistent.

(2) Consider the following constraint network.



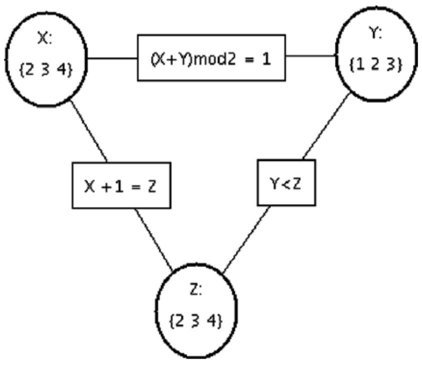
(a) Is this constraint network arc consistent?
(b) If it is, explain why the constraint network is arc consistent. If it isn't, make the network arc consistent and give all solutions.

(3) Consider the problem of scheduling each of four 1-h meetings starting at 2 pm, 3 pm or 4 pm. Let the scheduled start times for each meeting be A, B, C and D, respectively. The times must satisfy the following constraints: $A \neq B$, $C < A$, $A < D$, $B = D$, $C < B$ and $C < D$.



(a) Draw the constraints in the constraint graph.
(b) Make the network arc consistent and give all solutions.

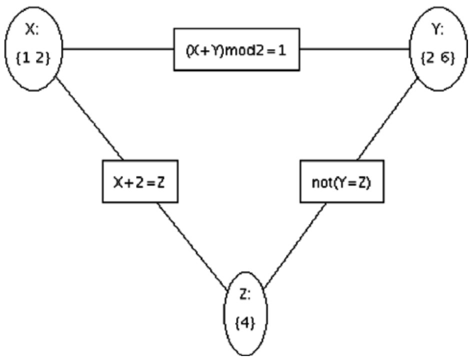
(4) Consider the following constraint network. Note that $(X + Y) \bmod 2 = 1$ means that $X + Y$ is odd.



(a) Is the constraint network arc consistent?
(b) If it is, explain why the constraint network is arc consistent. If not, make it arc consistent and show the consistent graph.
(c) Is domain splitting useful to solve this problem?
(d) If so, explain why and show all solutions. If not, explain why not.

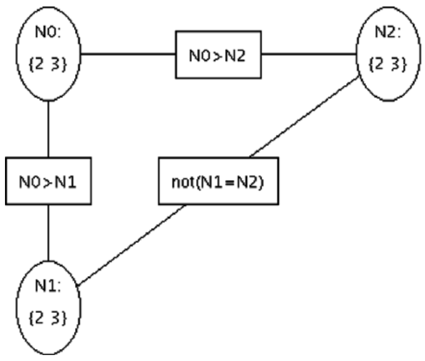
B.2. Post-test

(1) Consider the following constraint network. Note that $(X + Y) mod\ 2 = 1$ means that $X + Y$ is odd.



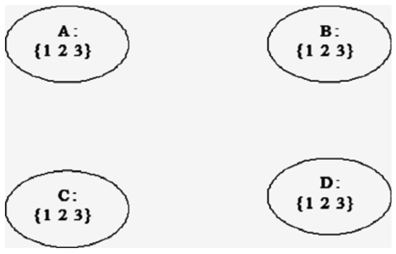
(a) Is this constraint network arc consistent?
(b) If it is, explain why the constraint network is arc consistent. If it isn't, state which arcs are not arc consistent and explain why the constraint network is not arc consistent.

(2) Consider the following constraint network.



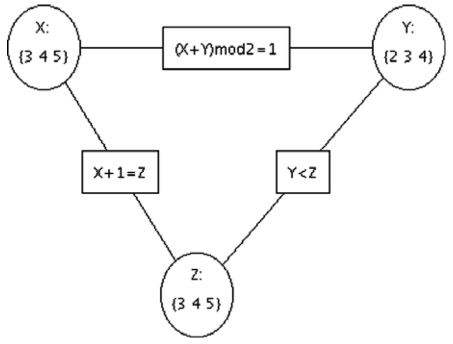
(a) Is this constraint network arc consistent?
(b) If it is, explain why the constraint network is arc consistent. If it isn't, make the network arc consistent and give all solutions.

(3) Consider the problem of scheduling each of four 1-h meetings starting at 1 pm, 2 pm or 3 pm. Let the scheduled start times for each meeting be A, B, C and D, respectively. The times must satisfy the following constraints: $A \neq B$, $C < A$, $A < D$, $B = D$, $C < B$ and $C < D$.



(a) Draw the constraints in the constraint graph.
(b) Make the network arc consistent and give all solutions.

(4) Consider the following constraint network. Note that $(X + Y) mod\ 2 = 1$ means that $X + Y$ is odd.



(a) Is the constraint network arc consistent?
(b) If it is, explain why the constraint network is arc consistent. If not, make it arc consistent and show the consistent graph.
(c) Is domain splitting useful to solve this problem?
(d) If so, explain why and show all solutions. If not, explain why not.

Appendix C. Questionnaires for pedagogical experiment 1

C.1. Non-applet group questionnaire

(1) How would you rate your level of confidence after the study on each of the topics below: (*circle a number for each topic*)

	Poor				Excellent
Variables	1	2	3	4	5
Variable domains	1	2	3	4	5
Constraints	1	2	3	4	5
Constraint satisfaction problem	1	2	3	4	5
The definition of arc consistency	1	2	3	4	5
Arc consistency algorithm AC-3	1	2	3	4	5
Domain splitting	1	2	3	4	5
Backtracking	1	2	3	4	5

(2) For the following statements, rate your agreement or disagreement: (*check a box for each row*)

Statement	Agree	Somewhat Agree	Neutral	Somewhat Disagree	Disagree
The paper sample problems helped me learn the material from the book					
The book alone would have been enough to learn the material					
It was difficult to follow the steps of the algorithm with the paper sample problems					
Seeing the network change at every step would have helped me study better					

(3) How much time do you think you spent figuring out the notation used in the sample problems? (*check a box*)
☐ *less than 5 min*
☐ *5–10 min*
☐ *10–20 min*
☐ *20–30 min*
☐ *over 30 min*

(4) The time given to read the book was: (*check a box*)
☐ *more than enough (if you check this state how long you spent reading)* ___________
☐ *enough*
☐ *barely enough*
☐ *not enough (if you check this state how long you think you needed)* ___________

(5) The time given to study the sample problems was: (*check a box*)
- ☐ *more than enough (if you check this state how long you think you needed)* ___________
- ☐ *enough*
- ☐ *barely enough*
- ☐ *not enough (if you check this state how long you think you needed)* ___________

C.2. Applet group questionnaire

(1) How would you rate your level of confidence after the study on each of the topics below: (*circle a number for each topic*)

	Poor				Excellent
Variables	1	2	3	4	5
Variable domains	1	2	3	4	5
Constraints	1	2	3	4	5
Constraint satisfaction problem	1	2	3	4	5
The definition of arc consistency	1	2	3	4	5
Arc consistency algorithm AC-3	1	2	3	4	5
Domain splitting	1	2	3	4	5
Backtracking	1	2	3	4	5

(2) For the following statements, rate your agreement or disagreement: (*check a box for each row)*

Statement	Agree	Somewhat Agree	Neutral	Somewhat Disagree	Disagree
The applet helped me learn the material from the book					
The book alone would have been enough to learn the material					
It was difficult to follow the steps of the algorithm with the applet					
Looking at examples worked out on paper would have helped me study better					

(3) How much time do you think you spent figuring out how to use the applet? (*check a box*)
- ☐ *less than 5 min*
- ☐ *5 to 10 min*
- ☐ *10–20 min*
- ☐ *20–30 min*
- ☐ *over 30 min*

(4) The time given to read the book was: (*check a box*)
- ☐ *more than enough (if you check this state how long you spent reading)* ___________
- ☐ *enough*
- ☐ *barely enough*
- ☐ *not enough (if you check this state how long you think you needed)* ___________

(5) The time given to study the sample problems was: (*check a box*)
- ☐ *more than enough (if you check this state how long you think you needed)* ___________
- ☐ *enough*
- ☐ *barely enough*
- ☐ *not enough (if you check this state how long you think you needed)* ___________

(6) For the following applet features, please check the features you feel helped you understand the material or work through the problems:

	Comments or suggestions for improvements
☐ The messages above the graph	______________________________
☐ The domain-splitting history area below the graph	______________________________
☐ The help pages	______________________________
☐ Being able to manually split domains by clicking on variables	______________________________
☐ Backtracking	______________________________
☐ Fine Step button	______________________________
☐ Clicking on arcs to make them consistent	______________________________
☐ Auto Arc-Consistency button	______________________________
☐ AutoSolve button	______________________________
☐ The colour changing of the arcs	______________________________

(7) For the following applet features, please check the features you feel hindered your learning, were not useful or were hard to use:

	Comments or suggestions for improvements
☐ The messages above the graph	______________________________
☐ The domain-splitting history area below the graph	______________________________
☐ The help pages	_______________________________
☐ Being able to manually split domains by clicking on variables	_______________________________
☐ Backtracking	_______________________________
☐ Fine Step button	_______________________________
☐ Clicking on arcs to make them consistent	_______________________________
☐ Auto Arc-Consistency button	_______________________________
☐ AutoSolve button	_______________________________
☐ The colour changing of the arcs	_______________________________

Appendix D. Questionnaires for pedagogical experiment 2

D.1. Questionnaire 1

(1) For the following statements, rate your agreement or disagreement and try to explain your answer : (*check a box for each row)*

Statement	Agree	Somewhat Agree	Neutral	Somewhat Disagree	Disagree
Using the applet helped me more than looking at the sample problems on paper					
Please Explain:					
I liked using the applet more then studying with the sample problems on paper					
Please Explain:					
Looking at the sample problems on paper helped me study better than the applet					
Please Explain:					
I liked studying with the sample problems on paper more then with the applet					
Please Explain:					

(2) For each pair of adjectives, check one box that reflects the extent to which you believe the adjectives describe the applet *(please read adjectives carefully)*.

Confusing	☐	☐	☐	☐	☐	☐	☐	☐	Clear
Boring	☐	☐	☐	☐	☐	☐	☐	☐	Exciting
Pleasing	☐	☐	☐	☐	☐	☐	☐	☐	Annoying
Unhelpful	☐	☐	☐	☐	☐	☐	☐	☐	Helpful

(3) For each pair of adjectives, check one box that reflects the extent to which you believe the adjectives describe the sample problems on paper *(please read adjectives carefully)*.

Confusing	☐	☐	☐	☐	☐	☐	☐	☐	Clear
Boring	☐	☐	☐	☐	☐	☐	☐	☐	Exciting
Pleasing	☐	☐	☐	☐	☐	☐	☐	☐	Annoying
Unhelpful	☐	☐	☐	☐	☐	☐	☐	☐	Helpful

(4) If you were taking a course to learn about CSPs, what would you like to use to study: (*check all that apply*)
- ☐ *CSP Applet*
- ☐ *Sample problems on paper*
- ☐ *Textbook*

(5) How much effort would you say you put in during this study: (*circle a number*)

None at all						A great deal
1	2	3	4	5	6	7

D.2. Questionnaire 2

(1) How would you rate your level of confidence after the study on each of the topics below: (*circle a number for each topic*)

	Poor				Excellent
Variables	1	2	3	4	5
Variable domains	1	2	3	4	5
Constraints	1	2	3	4	5
Constraint satisfaction problem	1	2	3	4	5
The definition of arc consistency	1	2	3	4	5
Arc consistency algorithm AC-3	1	2	3	4	5
Domain splitting	1	2	3	4	5
Backtracking	1	2	3	4	5

(2) The time given to study the Sample Problem 1 was: (*check a box*)
- ☐ *more than enough (if you check this state how long you think you needed)* ___________
- ☐ *enough*
- ☐ *barely enough*
- ☐ *not enough (if you check this state how long you think you needed)*___________

(3) The time given to study the Sample Problem 2 was: (*check a box*)
- ☐ *more than enough (if you check this state how long you think you needed)*___________
- ☐ *enough*
- ☐ *barely enough*
- ☐ *not enough (if you check this state how long you think you needed)*___________

(4) The time given to study the Sample Problem 3 was: (*check a box*)
- ☐ *more than enough (if you check this state how long you think you needed)* ___________
- ☐ *enough*
- ☐ *barely enough*
- ☐ *not enough (if you check this state how long you think you needed)*___________

(5) How much time do you think you spent figuring out how to use the applet? (*check a box*)
- ☐ *less than 5 min*
- ☐ *5 to 10 min*
- ☐ *10–20 min*
- ☐ *20–30 min*
- ☐ *over 30 min*

(6) For the following applet features, please check the features you feel helped you understand the material or work through the problems:

	Comments or suggestions for improvements
☐ The messages above the graph	______________________________
☐ The domain-splitting history area below the graph	______________________________
☐ The help pages	______________________________
☐ Being able to manually split domains by clicking on variables	______________________________
☐ Backtracking	______________________________
☐ Fine Step button	______________________________
☐ Clicking on arcs to make them consistent	______________________________
☐ Auto Arc-Consistency button	______________________________
☐ AutoSolve button	______________________________
☐ The colour changing of the arcs	______________________________

(7) For the following applet features, please check the features you feel hindered your learning, were not useful or were hard to use:

	Comments or suggestions for improvements
☐ The messages above the graph	______________________________
☐ The domain-splitting history area below the graph	______________________________
☐ The help pages	______________________________
☐ Being able to manually split domains by clicking on variables	______________________________
☐ Backtracking	
☐ Fine Step button	______________________________
☐ Clicking on arcs to make them consistent	______________________________
☐ Auto Arc-Consistency button	______________________________
☐ AutoSolve button	______________________________
☐ The colour changing of the arcs	______________________________

References

Adams, E.S., Carswell., L., Ellis, A., Hall, P., Kumar, A., Meyer, J., Motil, J., 1996. Interactive Multimedia Pedagogies, SIGCUE 24 (1–3), 182–191.

American Association for Artificial Intelligence, 2000. AI Topics. Available at: http://www.aaai.org/AITopics/html/welcome.html.

Amershi, S., Arksey, N., Carenini, G., Conati, C., Mackworth, A., Maclaren, H., Poole, D., 2005. Designing CIspace: pedagogy and usability in a learning environment for AI. ITiCSE 34, 178–182.

Amershi, S., Conati, C., 2006. Automatic Recognition of Learner Groups in Exploratory Learning Environments. ITS Journal, 463–472.

Atkinson, R.K., Derry, S.J., Renkl, A., Wortham, D., 2000. Learning from examples: instructional principles from the worked examples research. Review of Educational Research 70 (2), 181–214.

Baecker, R., 1981. Sorting Out Sorting (Videotape, 30 minutes), SIGGRAPH, Video Review 7.

Baecker, R., 1998. Sorting Out Sorting: A Case Study of Software Visualization for Teaching Computer ScienceSoftware Visualization: Programming as a Multimedia Experience. MIT Press, Cambridge, MA, pp. 369–381.

Ben-Ari, M., 1998. Constructivism in computer science education. SIGSCE 30 (1), 257–261.

Bergin, J., Brodlie, K., GoldWeber, M., Jimenez-Peris, R., Khuri, S., Patino-Martinez, M., McNally, M., Naps, T., Rodger, S., Wilson, J., 1996. An overview of visualization: its use and design. ITiCSE, 192–200.

Bloom, B.S., Krathwohl, D.R., 1956. Taxonomy of Educational Objectives; the Classification of Educational Goals, Handbook 1: Cognitive Domain. Addison-Wesley, Reading, MA.

Boroni, C.M., Goosey, F.W., Grinder, M.T., Lambert, J.L., Ross, R.J., 1999. Tying it all together creating self-contained, animated, interactive, web-based resources for computer science education. SIGCSE, 7–11.

Boroni, C.M., Goosey, F.W., Grinder, M.T., Lambert, J.L., Ross, R.J., 1998. A paradigm shift! The Internet, the web, browsers, java, and the future of computer science education. SIGCSE, 145–152.

Brown, A., 1987. Metacognition, executive control, self-regulation, and other more mysterious mechanisms. In: Weinert, F., Kluwe, R. (Eds.), Metacognition, Motivation, and Understanding. Lawrence Erlbaum Associates Inc., Hillsdale, NJ, pp. 65–116.

Brown, M., Meyrowitz, N., 1983. Personal computer networks and graphical animation: rationale and practice for education. SIGCSE, 296–307.

Brown, M.H., Sedgewick, R., 1984. A system for algorithm animation. Computer Graphics 18 (3), 177–186.

Byrne, M.D., Catrambone, R., Stasko, J.T., 1999. Evaluating animations as student aids in learning computer algorithms. Computers and Education 33 (5), 253–278.

Carlson, D., Guzdial, M., Kehoe, C., Shah, V., Stasko, J., 1996. WWW interactive learning environments for computer science education. SIGCSE, 290–294.

CIspace: tools for learning computational intelligence. Available at: http://www.cs.ubc.ca/labs/lci/CIspace/.

Cooper, C., 1997. Individual Differences. Oxford Illustrated Press, Oxford.

Cowley, B., Scragg, G., Baldwin, D., 1993. Gateway laboratories: integrated, interactive learning modules. SIGCSE, 180–184.

Crescenzi, P., Faltin, N., Fleischer, R., Hundhausen, C., Näher, S., Rößling, G., Stasko, J., Sutinen, E., 2002. The Algorithm Animation Repository. Program Visualization Workshop, 14–16.

Davis, S., Wiedenbeck, S., 2001. The mediating effects of intrinsic motivation, ease of use and usefulness perceptions on performance in first-time and subsequent computer users. Interacting with Computers 13, 549–580.

Demetriadis, S., Triatafillou, E., Pombortsis, A., 2003. A phenomenographic study of students' attitudes toward the use of multiple media for learning. ITiCSE, 183–187.

Dionne, M.S., Mackworth, A.K., 1978. ANTICS: a system for animating LISP programs. Computer Graphics and Image Processing 7 (1), 105–199.

Felder, R.M., 1993. Reaching the second tier – learning and teaching styles in college science education. Journal of College Science Teaching 23 (5), 286–290.

Fleischer, R., Kucera, L., 2001. Algorithm Animation for Teaching. In: Diehl, Stephan (Ed.), Software Visualization, State-of-the-Art Survey. Springer LNCS, Berlin, pp. 113–128.

Ghoniem, M., Fekete, J.-D., Castagliola, P., 2004. A comparison of the readability of graphs using node-link and matrix-based representations. IEEE Symposium on Information Visualization (InfoVis), 17–24.

Greiner, R., Schaeffer, J., 2001. The AIxploratorium: a vision for AI and the Web, IJCAI Workshop on Effective Interactive AI Resources.

Grissom, S., McNally, M.F., Naps, T., 2003. Algorithm visualization in CS education: comparing levels of student engagement. SOFTVIS, 87–94.

Gurka, J.S., Citrin, W., 1996. Testing effectiveness of algorithm animation. EEE Symposium on Visual Languages, 182–189.

Guzdial, M., Soloway, E., 2002. Teaching the Nintendo generation to program. Communications of the ACM 45 (4), 17–21.

Hansen, S., Narayanan, N.H., Hegarty, M., 2002. Designing educationally effective algorithm visualizations. Journal of Visual Languages and Computing 13 (3), 291–317.

Hansen, S., Narayanan, N.H., Schrimpscher, D., 2000. Helping learners visualize and comprehend algorithms. Interactive Multimedia Electronic Journal of Computer-Enhanced Learning 2 (1).

Hearst, M.A., 1994. Preface: improving instruction of introductory artificial intelligence, AAAI Fall Symposium on Improving the Instruction of Introductory AI, Technical Report FS-94-05, pp. 1–4.

Hubscher-Younger, T., Narayanan, N.H., 2003. Dancing hamsters and marble statues: characterizing student visualization of algorithms. Symposium on Software Visualization, 95–104.

Hundhausen, C.D., 1999. Toward effective algorithm visualization artifacts: designing for participation and communication in an Undergraduate Algorithms Course, Ph.D. Dissertation, Technical Report CIS-99-07, Department of Computer Science and Information Science, University of Oregon, Eugene.

Hundhausen, C.D., 2002. Integrating algorithm visualization technology into an Undergraduate Algorithms Course: ethnographic studies of a social constructivist approach. Computers and Education 39 (3), 237–260.

Hundhausen, C.D., Douglas, S.A., 2000. Using visualizations to learn algorithms: should students construct their own, or view and expert's?. IEEE Symposium on Visual Languages 21–28.

Hundhausen, C.D., Douglas, S.A., Stasko, J.T., 2002. A meta-study of algorithm visualization effectiveness. Journal of Visual Languages and Computing 13 (3), 259–290.

Ingargiola, G., Hoskin, N., Aiken, R., Dubey, R., Wilson, J., Papalaskari, M., Christensen, M., Webster, R., 1994. A repository that supports teaching and cooperation in the introductory AI course. SIGSCE 26 (1), 36–40.

Johnson-Laird, P.N., 1983. Mental models: towards a cognitive science of language, inference and consciousness. Cambridge University Press, Cambridge, UK.

Kehoe, C., Stasko, J., Taylor, A., 2001. Rethinking the evaluation of algorithm animations as learning aids: an observational study. International Journal on Human–Computer Studies 54 (2), 265–284.

Kehoe, C., Stasko, J., 1996. Using animations to learn about algorithms: an ethnographic case study, Technical Report GIT-GVU-96-20.

Knowlton, K., 1996. L6: Bell Telephone Laboratories low-level linked list language, 16 mm black and white file. Technical Information Libraries, Bell Laboratories Inc., Murray Hill, NJ.

Kolb, D., 1984. Experiential learning: experience as the source of learning and development. Prentice Hall, Englewood Cliffs, NJ.

Krebs, M., Lauer, T., Ottmann, T., Trahasch, S., 2005. Student-built algorithm visualizations for assessment: flexible generation, feedback and grading. ITiCSE, 281–285.

Large, A., Beheshti, J., Breuleux, A., Renaud, A., 1996. Effect of animation in enhancing descriptive and procedural texts in multimedia learning environment. Journal of the American Society for Information Science 47 (6), 437–448.

Lawrence, A.W., Badre, A., Stasko, J., 1994. Empirically evaluating the use of algorithm animations to teach algorithms. IEEE Symposium on Visual Languages, 48–54.

Levy, R.B., Ben-Ari, M., Uronen, P., 2003. The Jeliot 2000 program animation system. Computers and Education 40 (1), 1–15.

Manaris, B., Russell, I., 1996. AI Education Repository, Available at: http://www.cs.cofc.edu/~manaris/ai-education- repository/.

Mayer, R., 1981. The psychology of how novices learn computer programming. Computing Surveys 13 (1), 121–141.

MIT OpenCourseWare, Artificial Intelligence Tools, Available at: http://ocw.mit.edu/OcwWeb/Electrical-Engineering-and-Computer-Science/6-034Artificial-IntelligenceFall2002/Tools/, 2002.

Naps, T., Bergin, J., Jimenez-Peris, R., McNally, M., Patino-Martinez, M., Proulx, V., Tarhio, J., 1997. Using the WWW as the Delivery Mechanism for Interactive, Visualization-Based Instructional Modules, ITiCSE Working Group on Visualization, pp. 13–26.

Naps, T., Rodger, S., Rößling, G., Ross, R., 2006. Animation and visualization in the curriculum: opportunities, challenges, and successes. SIGCSE Panel Session, 328–329.

Naps, T.L., Cooper, S., Koldehofe, B., Leska, C., Rößling, G., Dann, W., Korhonen, A., Malmi, L., Rantakokko, J., Ross, R.J., Anderson, J., Fleischer, R., Kuittinen, M., McNally, M., 2003. Evaluating the educational impact of visualization. ITiCSE, 124–136.

Naps, T.L., Eagan, J.R., Norton, L.L., 2000. JHAVE – an environment to actively engage students in web-based algorithm visualizations. SIGCSE, 109–113.

Naps, T.L., Rodger, S., Velzquez-Iturbide, J., Rößling, G., Almstrum, V., Dann, W., Fleisher, R., Hundhausen, C., Korhonen, A., Malmi, L., McNally, M., 2002. Exploring the role of visualization and engagement in computer science education. ITiCSE, 131–152.

Norman, D.A., 1983. Some observations on mental models. In: Gentner, Dedre, Stevens, Albert L. (Eds.), Mental Models. Lawrence Erlbaum Associates, Hillsdale, NJ, pp. 7–14.

Paivio, A., 1971. Imagery and Verbal Processing. Holt, Rinehart & Winston, New York.

Paivio, A., 1983. The empirical case for dual coding. In: Yuille, J.C. (Ed.), Imagery, Memory, and Cognition: Essays in Honor of Allan Paivio. Lawrence Erlbaum Associates, Hillsdale, NJ.

Pane, J.F., Corbett, A.T., John, B.E., 1996. Assessing Dynamics in Computer-Based Instruction. SIGCHI '96, 197–204.

Plaisant, C., Rose, A., Rubloff, G., Salter, R., Shneiderman, B., 1999. The Design of History Mechanisms and their Use in Collaborative Educational Simulations. Computer Support for Collaborative Learning, 348–359.

Poole, D., Mackworth, A., 2001. CIspace: Tools for Learning Computational Intelligence. IJCAI Workshop on Effective Interactive AI Resources.

Poole, D., Mackworth, A., Goebel, R., 1998. Computational Intelligence: A Logical Approach. Oxford University Press, New York.

Price, B.A., Baecker, R.M., Small, I.S., 1993. A principled taxonomy of software visualization. Journal of Visual Languages and Computing 4 (3), 211–266.

Rantakokko, J., 2004. Algorithm visualization through animation and role plays. Program Visualization Workshop 407, 76–81.

Riding, R., Rayner, S., 1998. Cognitive Styles and Learning Strategies. David Fulton Publishers, London.

Rieber, L., 1989. The effects of computer animated elaboration strategies and practic on factual and application learning in an elementary science lesson. Journal of Educational Computing Research 5 (4), 431–444.

Rieber, L., 1990. Animation in computer-based instruction. Educational Technology Research and Development 38 (1), 77–86.

Rößling, G., Naps, T.L., 2002. A testbed for pedagogical requirements in algorithm visualizations. SIGCSE 34 (3), 96–100.

Russell, S., Norvig, P., 2003. Artificial Intelligence: A Modern Approach, second ed. Prentice Hall, Englewood Cliffs, NJ.

Saraiya, P., Shaffer, C.A., McCrickard, D.S., North, C., 2004. Effective features of algorithm visualizations. SIGCSE, 382–388.

Schneiderman, B., 2003. Promoting universal usability with multi-layer interface design. ACM Conference on Universal Usability, 1–8.

Soloway, E., 1991. How the Nintendo generation learns. Communications of the ACM 34 (9), 23–28.

Stasko, J., 1990. Tango: a framework and system for algorithm animation. IEEE Computer 23 (9), 27–39.

Stasko, J., Badre, A., Lewis, C., 1993. Do algorithm animations assist learning? An empirical study and analysis. INTERCHI 93, 61–66.

Stasko, J., Hundhausen, C.D., 2004. Algorithm visualization. In: Fincher, S., Petre, M. (Eds.), Computer Science Education Research. Taylor & Francis, London, pp. 199–228.

Stern, L., Markham, S., Hanewald, R., 2005. You can lead a horse to water: how students really use pedagogical software. ITiCSE, 246–250.

Tversky, B., Morrison, J.B., 2001. The (in)effectiveness of animation in instruction. SIGCHI '01, 377–378.

Tversky, B., Morrison, J.B., Betrancourt, M., 2002. Animation: can it facilitate? International Journal of Human–Computer Studies 57, 247–262.

van Lehn, K., 1998. Analogy events: how examples are used during problem solving. Cognitive Science 22 (3), 347–388.

West, T.G., 1992. Visual thinkers, mental models and computer visualization. In: Cunningham, S., Hubbold, R.J. (Eds.), Interactive Learning through Visualizations. Springer-Verlag, Berlin, pp. 93–102.

Wilson, J., Katz, I.R., Ingargiola, G., Aiken, R., Hoskin, N., 1995. Students' use of animations for algorithm understanding. CHI, 238–239.