COSC460 Honours Report: Investigating the effects of input method on student learning in an ITS

November 12, 2009

Aidan Bebbignton
ajb289@student.canterbury.ac.nz
Department of Computer Science and Software Engineering
University of Canterbury, Christchurch, New Zealand

Supervisor: Professor Antonija Mitrović
Tanja.Mitrovic@canterbury.ac.nz

Abstract

Personalised one-to-one tutoring is known to be the most effective form of instruction. However, with limited resources, the situation in most educational settings falls far short of this ideal. Intelligent Tutoring Systems (ITSs) are computerised systems which have the potential to provide instruction which is pedagogically equivalent to personalised tutoring. Extensive research has been focused on improving the effectiveness of ITSs in a variety of ways. This report presents an evaluation of the effectiveness for learning of two different interaction styles: selection and typing.

Two interfaces for an ITS for thermodynamics called Thermo-Tutor are designed and developed. The designs are justified in terms of theory from psychology and cognitive science. Finally an evaluation is performed to measure the relative effectiveness of these interfaces and these results are analysed and discussed.

Thermo-Tutor is developed using ASPIRE; a general purpose authoring system for constraint-based ITSs. ASPIRE is used to reduce the development time and evaluate its effectiveness. The effectiveness of ASPIRE is also analysed and discussed.

The results related to the relative effectiveness of the two interfaces are inconclusive. However, results show that students did learn while using Thermo-Tutor and comments from students were generally positive. Therefore, development of Thermo-Tutor with ASPIRE was successful. This research also helped to identify some possible improvements for the future.

Acknowledgments

Special thanks to my supervisor, Tanja Mitrović for ongoing guidance, inspiration and reassurance throughout this project and others. Thank you to all fellow members of ICTG especially Moffat Mathews, Jay Holland and Amali Weerasinghe for guidance both technical and otherwise. Thank you to all the honours students for being in the same boat. Finally, a special thanks to Jasmine Morgan for support and generally holding my life together!

Contents

1 Introduction **1**
- 1.1 Learning in Modern Society . . . 1
- 1.2 The State of Education . . . 1
- 1.3 Computers in Education . . . 1
- 1.4 Overview . . . 2
- 1.5 Motivations for this Research . . . 2
- 1.6 Structure of this Report . . . 2

2 Related Work **3**
- 2.1 Intelligent Tutoring Systems . . . 3
  - 2.1.1 Modelling . . . 3
  - 2.1.2 ASPIRE . . . 4
  - 2.1.3 Thermo-Tutor . . . 6
- 2.2 Learning, Memory and Cognition . . . 12
  - 2.2.1 Generation Effect . . . 12
  - 2.2.2 Cognitive Load Theory . . . 13
  - 2.2.3 Guidance and Assistance Dilemma . . . 13

3 Design and Implementation **15**
- 3.1 Goals and Hypotheses . . . 15
  - 3.1.1 Research Objectives . . . 15
  - 3.1.2 Goals . . . 15
  - 3.1.3 Hypotheses . . . 15
- 3.2 Completing Thermo-Tutor . . . 16
- 3.3 Control Interface . . . 17
- 3.4 Experimental Interface . . . 17
  - 3.4.1 Calculating Unknowns . . . 18
- 3.5 Motivation for Redesign . . . 19
- 3.6 Design Tradeoffs . . . 20
- 3.7 Added Complications . . . 21
- 3.8 Tools Used . . . 22

4 Evaluation **23**
- 4.1 Description . . . 23
- 4.2 Participants . . . 23
- 4.3 Evaluation Plan . . . 23
- 4.4 Revised Evaluation Plan . . . 23
- 4.5 Technical Difficulties . . . 24
- 4.6 Results . . . 24
  - 4.6.1 Student Performance . . . 24
  - 4.6.2 Questionnaire . . . 25
  - 4.6.3 Learning Curves . . . 26

5 Discussion and Future Work **28**
5.1 Possible Improvements 28
5.2 ASPIRE 29
5.2.1 Generating the Domain Model 29
5.2.2 Causes of these Issues 30
5.2.3 ASPIRE's Prospective Audience 31

6 Conclusions **32**

Bibliography **35**

A Tests and questionnaire **36**

1 Introduction

1.1 Learning in Modern Society

Learning has always been fundamental to human existence and continues to be crucial in modern society. The importance of learning in modern society also looks likely to increase. Learning is a process which changes the way we are likely to behave. The experience of learning increases the likelihood that we will react in some way to a given stimuli [1]. Learning has a profound influence over the human experience. It influences the way we perceive our world and the way we interpret new information. Consequently, it also influences how we react to new situations and information. Learning influences practically every aspect of our lives.

Learning has been recognised as important across practically all areas of our society including business, government and military. Many organisations and individuals are seeking more effective learning methods to reap the many advantages that learning offers. Organised learning is often undertaken to increase productivity and efficiency, achieve a variety of performance-oriented goals, and gain and retain an advantage over the competition. Consequently, learning has been studied in many fields, and many theories have been proposed to attempt to explain and understand the processes involved in learning. These theories include behaviourism, cognitivism, and constructivism. Learning has also been studied from a physiological point of view in an attempt to explain the process of learning in a more scientific manner [2]. However, despite extensive research many disagreements among experts in the field remain.

1.2 The State of Education

Studies have shown that the best student learning occurs during one-on-one interaction with an experienced tutor [3, 4]. However the current situation in education falls a long way short of this ideal. Even in well-funded institutions the student-to-teacher ratio is not sufficient to make this feasible, and the problem is even more prevalent in poorly funded institutions. Because of this, teachers are often forced to make their teaching technique suitable for the average student. High achieving students become bored and frustrated by the slow pace and lack of challenge and so fail to achieve their best. Low achieving students find the work very difficult and never receive the level of attention they require to achieve their best. Furthermore, even those students who find the difficulty level appropriate are not ideally accommodated by such a system because they do not receive the amount of one-on-one interaction with the tutor that they require for optimal learning.

1.3 Computers in Education

Computers have the potential to solve some of these problems in modern education systems. Computer-based educational systems could provide the level of scalability that is needed to educate on a large scale with the current human and financial resource limitations. Computer-based educational systems could provide low operating costs and a relatively low initial investment if they are deployed on a suitable scale.

Some approaches to computer-based education include computer-based learning or training, computer-supported collaborative learning, learning management systems, computer-aided instruction and intelligent tutoring systems. These examples are often placed in the category of "E-Learning". The exact definition of E-Learning is not clear, and it is a term often associated with distance learning. To prevent confusion, I will continue to use the term Computer-Based Education to describe any computer system which employs "pedagogy empowered by digital technology" [5]. These systems do not necessarily need to be used in the

context of distance learning, or any other specific context. The only requirement is that they use computers to support some pedagogy.

Computer-Based Education (CBE) has been the subject of much excitement in the past two decades and also much criticism in more recent years [6, 7]. Those who are critical of CBE generally claim that it has failed to live up to promises. Those who support CBE say it has great potential to improve the quality of teaching and learning, but the implementation is still in the first phase.

There is a general opinion among some experts that the majority of CBE applications are too simple. Many CBE applications provide an experience which is inferior to more traditional methods. Many are in their nature very similar to traditional methods which have been known to be ineffective in some way. For example, learning management systems can be pedagogically equivalent to a notice board, and computer-based instruction can be pedagogically equivalent to textbooks. The potential advantages of CBE systems will not be fully realised until they provide some pedagogical benefit to the student. The technology in itself is of very limited benefit; what counts is how this technology is applied.

1.4 Overview

An Intelligent Tutoring System (ITS) is any computer system that provides individualised, adaptive instruction to students. ITSs aim to provide an experience that is similar to personal tutoring without the need for human intervention. The primary mechanism used by ITSs to provide adaptiveness is student modelling. The exact method used for student modelling varies between ITSs. The fundamental requirement of a student model is that it contains a model of the student's knowledge that can be used to influence pedagogical decisions. Student modelling techniques are usually based on theory from psychology and cognitive science. Applying them in practice often requires the use of techniques from the field of artificial intelligence. Consequently, ITSs are among the most sophisticated forms of CBE in widespread use today. The benefits of ITSs and other sophisticated forms of CBE have been demonstrated in numerous studies [8, 9, 10].

However, many open questions remain regarding how these systems should operate and the experience they should provide to the student. For example, there is much uncertainty regarding the amount of assistance that ITSs should provide. Extensive research has been conducted to answer similar questions in the broader field of education [11], and in ITSs specifically [12]. However, assistance can be provided in many different ways, and it is not always clear how results from the field of education should be applied to CBE. Moreover, much of the theory which was obtained via fairly artificial psychological experiments is only beginning to be applied and tested in more realistic educational scenarios. This report presents an attempt to apply some of this previous research to an ITS for introductory thermodynamics called Thermo-Tutor. The theory is discussed and used to guide the redesign of the interface for Thermo-Tutor. Finally, an evaluation is conducted to identify the benefits and costs of this new design.

1.5 Motivations for this Research

Although ITSs have been shown to achieve high learning rates in a short amount of time in a variety of domains [8, 9, 10], their performance still falls significantly short of personal human tutoring [4]. An on going goal of ITS development is to provide more effective learning. There are many ways that researchers are seeking to achieve this goal. This research investigates the interaction style of the interface as one possible way of providing more effective learning in an ITS.

1.6 Structure of this Report

Section 2 summarises and discusses the background material relevant to this project. It provides an overview of ITSs, theories of cognition, and other effects that may be observable in this research. It is not intended to provide an exhaustive analysis of this material. I will outline only the most immediately relevant concepts and provide references for further reading. Section 3 outlines the goals and hypotheses for this research. Section 4 describes the design and implementation of Thermo-Tutor and the alterations made to the interface. The motivations behind the redesign, the relevant forces and their associated trade-offs are also briefly discussed. Section 5 describes the evaluation and discusses the results. Section 6 provides an overall discussion of this research and possible future work. Finally, Section 7 summarises and concludes this report.

2 Related Work

2.1 Intelligent Tutoring Systems

Intelligent Tutoring Systems aim to emulate one-on-one interaction in order to achieve results similar to personal human tutoring. In addition, ITSs have the potential to alleviate the issues found in the education system associated with a lack of resources. After an ITS has been developed it can be used to teach any number of students at very low cost. In order for an ITS to effectively emulate one-on-one tutoring it must fulfil two main requirements. Firstly, it must be able to adapt to accommodate students of varying levels of ability. Secondly, it must be able to give meaningful feedback to the student regarding their errors to help them correct their misconceptions.

There have been several approaches devised to allow an ITS to adapt to students of varying ability levels. The ITS must maintain some model of each student's knowledge; this is known as a student model. Several student modelling approaches have been devised, each with strengths and weaknesses. Student modelling is fundamental to the operation of an ITS. A suitable student model would usually include information regarding what the student has understood, misunderstood and not yet encountered. The degree of adaptability varies between ITSs. Usually ITSs are adaptive in that they are capable of selecting a problem of a suitable difficulty for a given student.

An ITS must also be capable of providing meaningful feedback to the student regarding their errors. To achieve this, the ITS must have some mechanism of evaluating students' solutions and/or tracking their progress. Student modelling approaches are described in greater detail in the following section. An ITS can also adapt the type and amount of feedback given to students based on their ability. This would usually mean that low ability students are given more help than high ability students.

To effectively emulate the behaviour of a personal tutor also requires that the ITS include some model of the domain it is designed to teach. Several modelling techniques have been proposed to support this. The modelling technique used is often considered a major defining characteristic of an ITS. Today the two most common modelling techniques are Model Tracing (MT) and Constraint-Based Modelling (CBM), which are used by Cognitive Tutors and Constraint-Based Tutors respectively. The student modelling technique typically corresponds closely to the domain modelling technique. Below I give a description of Cognitive Tutors and Constraint-Based Tutors, see [13] for a more detailed comparison of the two techniques.

2.1.1 Modelling

Cognitive Tutors

Cognitive tutors utilize the MT technique, which is derived from the ACT-R cognitive theory [14]. This theory assumes there to be two long-term memory states: declarative and procedural. The theory explains that the human learner goes through several stages of knowledge construction. The learner firstly gains declarative knowledge, which includes factual details, theorems and other principles. This declarative knowledge is later organized into procedural knowledge, which is goal-oriented. Procedural knowledge is more efficient to use and less error-prone because it leaves fewer decisions to make while performing some procedure. Declarative knowledge can be quite detailed, but it lacks the structure, organization and goal-oriented nature of procedural knowledge.

Procedural knowledge is defined in terms of production rules, each of which define what action should be taken, given some circumstance. The fundamental assumption of ACT-R is that a person's cognitive skills in a domain are defined by the production rules they possess. Therefore, the goal of cognitive tutors

is to teach the student the correct set of production rules. So clearly the mechanism used by cognitive tutors to evaluate students' solutions and provide feedback (the domain model) should also be defined in terms of production rules.

Cognitive tutors model domain expertise as a set of production rules, which is also known as a generic student model. This cognitive model describes the domain expertise required to perform tasks well (and perhaps also poorly). Cognitive tutors react immediately to each step a student takes. An error is defined as when the student's action does not match any rule, or when it does match one of the buggy rules. This process is called model tracing because it involves tracing the student's actions against the expected actions defined by the generic student model. Cognitive tutors must also maintain a student-specific model, which is constructed through a process called knowledge tracing. This student-specific model (also known as the student model) is the tutor's representation of the student's knowledge.

Each production rule in a cognitive tutor is defined as follows:

```
If goal is <goal>
And the situation is <situation>
Then perform the action <action>
```

For example:

```
If the goal is 'to have a room temperature of 21 degrees'
And the situation is 'the current room temperature is below 21 degrees'
Then perform the action 'turn on the heater'.
```

Constraint-Based Tutors

CBM was originally proposed by Ohlsson [15] as a method which may overcome the intractable nature of student modelling. CBM is rooted in Ohlsson's own theory of learning from performance errors. This theory points out that even when we have been taught how to perform some task we often continue making mistakes for some time. According to this theory, we continue making mistakes when we have not yet internalized our declarative knowledge as procedural knowledge. This means we must make many more decisions while performing some procedure which increases the likelihood of errors. With more practice we can identify the mistakes we make (perhaps with the aid of a mentor) and modify our procedural knowledge accordingly.

CBM is an inherently simpler mechanism that MT. The Constraint-Based approach aims to determine the validity of the state the student is currently in. It does not aim to immediately determine if the student has left the path of the problem solving procedure because it deems this to be unnecessary. To put this differently, "In CBM, we are not interested in what the student has done, but in what state they are currently in" [13]. This simplicity is justified by recognizing that the student cannot arrive at a correct solution by traversing a problem state that violates any fundamental principle of the domain. Thus, by determining that the state the student is currently in is valid, we will have determined that the problem solving procedure used by the student is also valid.

Each constraint in a Constraint-Based tutor is defined as follows:

```
If <relevance condition> is true then
<satisfaction condition> must also be true
```

For example:

```
If 'the current room temperature is below 21 degrees' is true then
'the heater is on' must also be true.
```

2.1.2 ASPIRE

Description and Purpose

Efforts to reduce the cost of developing ITSs began with ITS shells such as WETAS; a web-based shell which facilitates the development of constraint-based tutors [16]. ITSs typically share a common architecture in the way they are composed of modules. ITS shells typically provide all domain independent

modules. While many modules in an ITS are domain independent, one particular component is not; the domain model. The domain model also consumes the most time and requires the most expertise during development. ASPIRE is an authoring system for the development of constraint-based ITSs. The goal of ASPIRE is to accelerate the development of the domain model to reduce the overall cost of developing constraint-based ITSs [17]. ASPIRE guides the author through a semi-automated process to develop the domain model. It then allows the author to deploy their tutor on the web, with all other necessary modules included automatically. It is also hoped that ASPIRE will allow a domain expert with little or no programming background to create an ITS for their domain. This will help alleviate many common issues encountered in ITS development such as the author failing to properly understand the domain, and communication failures between the domain expert and the author.

In Constraint-Based Modelling (CBM) the single biggest task is creating the constraints that make up the knowledge base. This task can seem even greater to a domain expert with little programming expertise. ASPIRE can generate many of these constraints automatically, in a way that domain experts with little or no programming background could carry out. Evaluation has shown [17] that ASPIRE can generate all syntax constraints that were developed manually for KERMIT [18, 19], and all but two that were developed for NORMIT [18, 20]. It was also found that ASPIRE could generate 85% of the semantic constraints found in KERMIT, and the complete set of semantic constraints that exist for NORMIT. The difference between syntax and semantic constraints is discussed in the following section.

Authoring Process

The main purpose of ASPIRE is to simplify the process of authoring an ITS. It does so by allowing the author to describe the domain in an abstract manner, without the need for programming. The following steps make up ASPIRE's authoring process [17].

1. Specifying the domain characteristics
2. Composing the domain ontology
3. Modelling the problem and solution structures
4. Designing the student interface
5. Adding problems and solution
6. Generating syntax constraints
7. Generating semantic constraints
8. Validating the generated constraints
9. Deploying the tutoring system

I will give an outline of what is involved in developing the ontology (step 2) and adding problems and solutions (step 5) because these steps have a direct effect on the generation of syntax and semantic constraints. For a more complete description of the authoring process see [17].

The ontology is an abstract model of the concepts in the domain. The following general definition has been given [21]:

"An ontology may take a variety of forms, but necessarily it will include a vocabulary of terms, and some specification of their meaning. This includes definitions and an indication of how concepts are interrelated which collectively impose a structure on the domain and constrain the possible interpretations of terms."

The ontology that ASPIRE allows an author to create is simpler than some other ontologies. This is necessary for it to be easily understood by domain experts with little domain modelling experience. ASPIRE provides a workspace for the author to visually develop the ontology. Creating the ontology in ASPIRE is not unlike creating a UML class diagram. The author specifies the hierarchical structure of concepts in terms of sub- and super-concepts. They can also define various properties of concepts and other relationships which are not of a hierarchical nature. Studies have shown ontologies to be an effective way of expressing domain concepts [22]. A preliminary study has also shown that an ontology assists in the composition of constraints for constraint-based tutors [23]. The study showed that authors reflect on the domain, organize constraints and produce more complete constraints for a domain when using an ontology.

ASPIRE uses the ontology created by the author to generate syntax constraints. These constraints check that the student's solution is syntactically valid, but not necessarily correct. To generate the syntax constraints, ASPIRE examines the relationships and properties defined in the ontology, as well as the restrictions placed on these relationships and properties.

Before ASPIRE can generate semantic constraints the author must define a set of problems and solutions for the domain. For each problem the author must specify the problem statement and one or more correct solutions. The author can specify multiple correct solutions for a single problem, and ASPIRE will use this to learn about the different ways of solving that problem. Semantic constraints are generated by putting the problems and their solution sets through a machine learning algorithm [17]. This algorithm analyses solutions pair-wise to determine similarities and differences between them. The resulting constraints compare the student's solution to the correct solutions defined for a problem to identify errors and provide feedback to help the student reach the correct solution.

For this report it is also important to understand the process of validating the generated constraints (step 8). It is always necessary to check that these constraints correctly catch all errors and provide appropriate feedback. The degree to which the constraints need to be altered will vary depending on the domain. It hoped that no significant alterations will be required. However, it is expected that at least the feedback messages of all constraints will have to be altered. ASPIRE generates default feedback messages, but these will usually have to be rewritten to be easier to comprehend.

For more complex domains it is expected that more substantial alterations to the constraints will be required. Usually these changes will be made using the same constraint language which ASPIRE uses in its own generated constraints. This language is a subset of Common Lisp, and includes many useful functions to make expressing constraints as simple as possible. More sophisticated functionality can also be implemented with domain functions. Domain functions give the developer the full power of Common Lisp, allowing constraints of arbitrary complexity to be written. ASPIRE does not generate constraints which use domain functions, and it is not expected that a domain expert with no programming expertise would be capable of writing domain functions.

2.1.3 Thermo-Tutor

Description and Purpose

Thermo-Tutor is intended to teach introductory thermodynamics to second year engineering students. More specifically, it is intended to teach thermodynamic cycles in closed systems. It is not intended to cover the entire course, nor the homework component of an entire course. Instead, it is intended to complement a particularly troublesome section of the course and to help get students started in thermodynamics.

Several people have contributed to the development of Thermo-Tutor. This project follows from a summer project I conducted to continue development of Thermo-Tutor [24].

Thermo-Tutor is being developed using ASPIRE in order to reduce the development time required and to evaluate ASPIRE's performance.

Solving Thermodynamics Problems

When designing the interface for Thermo-Tutor one of the goals was to make the students' experience correspond closely to how they would usually solve such problems on paper. This is important to make it natural and easy to use, and also to support transfer of the problem solving skills students gain from using the tutor to solving similar problems outside the tutor. We aimed to make Thermo-Tutor suitable for solving problems related to cycles in closed systems. To solve such problems usually involves two main phases:

1. **Draw the diagram:** This includes collating all that is known about the cycle. From this information the student draws a diagram that represents all the states and transitions in the cycle.
2. **Calculate unknown properties:** Using a number of formulae it is possible to calculate the unknown properties of states and transitions.

The first phase in solving a thermodynamic cycle problem is to draw a state diagram. This shows the states and transitions involved in the cycle. A state is a point where the system is at equilibrium under a

given set of conditions. A state is described by three properties; pressure, volume and temperature. The relationship between these properties is also defined such that if two properties are known then the third can be calculated without any additional information. A transition (also known as a process) describes the way the system changes from an initial state to a final state. Transitions are generally distinguished from others by what remains constant throughout the transition (pressure, volume, temperature or heat).

A diagram of a thermodynamic cycle shows the states involved in the cycle with axis of pressure versus volume (see Figure 2.2). Often this diagram will not include scales on the axis as it is used primarily to show relative positions of states.

The diagram also shows the transitions between these states. The shape and direction of these transitions gives some indication as to the type of transition. In Figure 2.2 it is clear that transition 2 - 3 is an isochoric (constant volume) transition as it is perpendicular to the x-axis.

When the diagram is complete the next phase is to build upon what is known about the cycle until all properties of all transitions and states are known. There are a quite a number of formulae used during this phase and there are usually many ways of determining each property. It is often necessary to calculate certain properties in some order, but there are also usually several properties which can be calculated at any one point in the problem solving process. The set of properties which can be calculated is dependent on the properties that are currently known, and the types of transitions present in the cycle. For example, the change in internal energy of a transition can often not be calculated until the temperatures at both the initial and final states are known, and some formulae are only valid for isothermal (constant volume) transitions. In general, it is necessary to determine at least two properties of each state, and from this it is possible to determine all properties of transitions. However, it is sometimes sufficient to determine just one property of some states. This is because some formulae which are related to specific types of transitions implicitly contain information about these types of transitions.

Problem Set and Scope

Thermo-Tutor is designed to support problems involving thermodynamic cycles in closed systems. This restriction clearly defines the scope of supported problems and also means that more advanced concepts of thermodynamics do not need to be considered. In particular this means that all problems only require students to have an understanding of the first law of thermodynamics.

Thermo-Tutor is not intended to cover an entire course, nor the homework component of an entire course. It is intended to give new students the opportunity to experiment with the fundamentals of thermodynamics. The common errors which new students make appear even in the relatively simple problems that Thermo-Tutor supports. Often these early misconceptions are held for some time, and the problems caused by these misconceptions are exasperated when the student attempts to solve more difficult problems. Obviously providing an environment where early misconceptions can be corrected would be of great benefit to both student and teacher.

Student Interface

The problems that Thermo-Tutor is designed to support involve three steps. An additional step is included after drawing the diagram so all the information required for calculating unknown properties is visible on screen.

1. **Draw the diagram:** Draw a diagram showing all the defined states for a given cycle, as well as all the transitions between those states. While constructing the diagram the student also defines all known properties of states and transitions that are specified or can be easily derived from the problem statement.
2. **Specify the constants:** Specify five constants which will be necessary to calculate unknown properties.
3. **Calculate unknown properties:** Calculate the remaining unknown properties of states and transitions using a set of thermodynamic formulae and principles.

Figure 2.1 shows the overall layout of the interface. The interface includes an area for the problem statement (1), feedback (2) and solution workspace (3). ASPIRE provides the areas for the problem state-

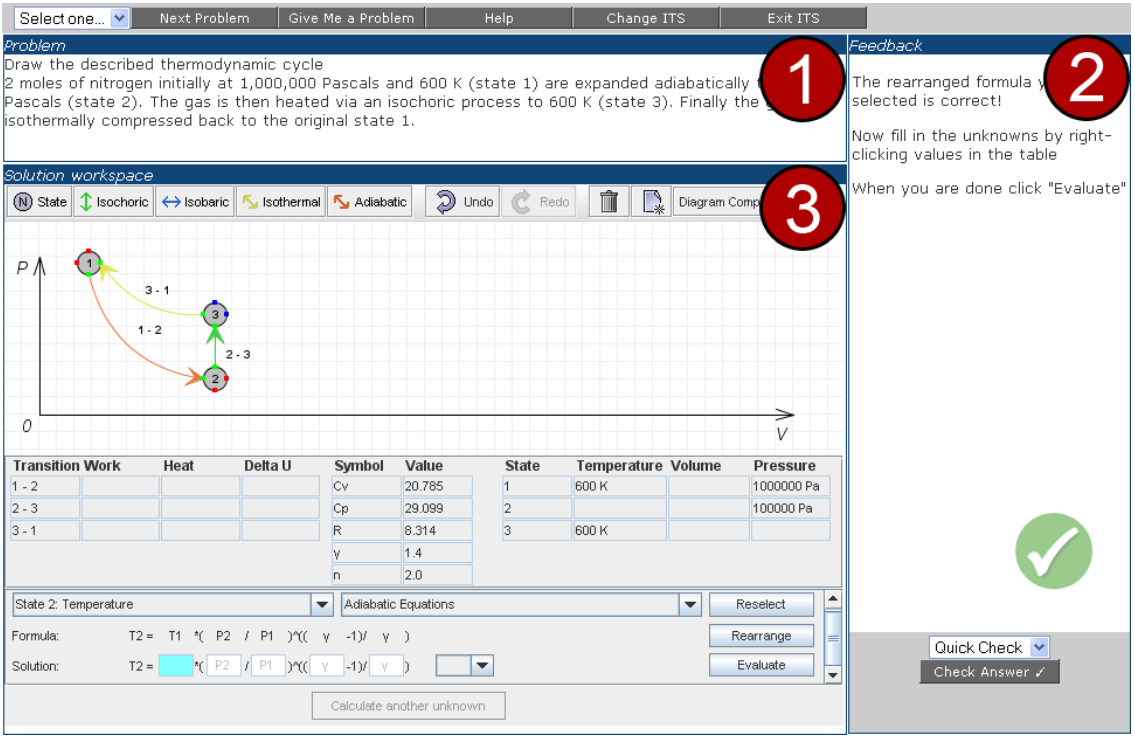


Figure 2.1: Thermo-Tutor Interface Layout

ment (1) and feedback (2), and also the controls above the problem statement. The solution workspace (3) is where the student solves problems. For Thermo-Tutor, it is implemented as a Java applet. The remainder of this section is concerned with the way the student uses this Java applet to solve thermodynamics problems.

Figure 2.2 shows the overall layout of the solution workspace. The solution workspace includes a toolbar (1), diagram workspace (2), data summary (3) and formula workspace (4). The toolbar (1) allows the user to select objects to place on the diagram workspace, as well as other functions such as undo/redo, deleting objects, clearing the diagram workspace and submitting a solution. The diagram workspace (2) is where the user draws the states and transitions to represent the thermodynamic cycle. The data summary (3) displays all the information known relating to the thermodynamic cycle. The formula workspace (4) is where the user calculates unknown values to iteratively build upon what is known about the cycle. The values of unknowns appear in the data summary when they have been calculated. Double-clicking or right-clicking values on the data summary causes them to be copied into the currently selected cell on the formula workspace (highlighted blue above).

Drawing the Diagram

Thermo-Tutor allows the student to create a diagram of a thermodynamic cycle with a simple point-and-click style interface. At the same time the student can enter any known properties of states and transitions provided in the problem statement. Figure 2.3 shows the problem workspace when the user has just added a new state to the diagram workspace. Some properties of states may not be explicitly given by the problem statement, but instead must be inferred by the student using their knowledge of thermodynamics. For example, an isobaric transition is one in which the pressure remains constant. In this case the student may only be given the pressure of the start state, the student would then be required to infer that the pressure of the end state is the same. During this first step the student will not be required to carry out any calculations. All the information required to complete this step comes directly from the problem statement.

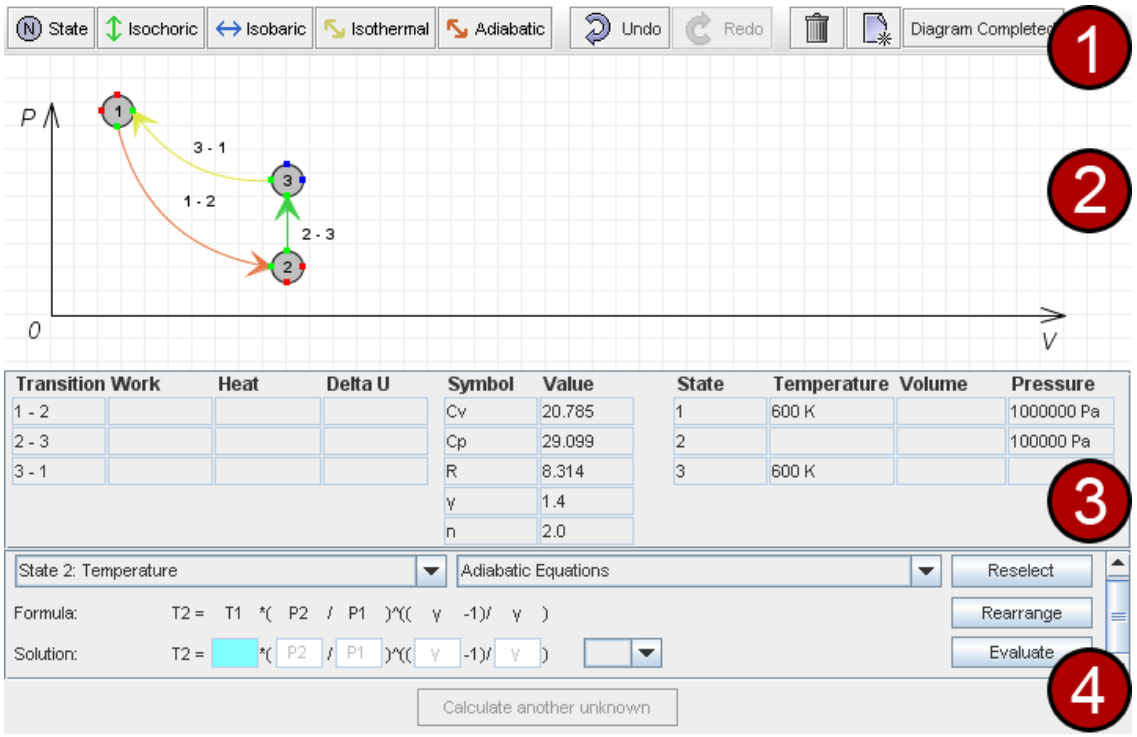


Figure 2.2: Layout of the Solution Workspace

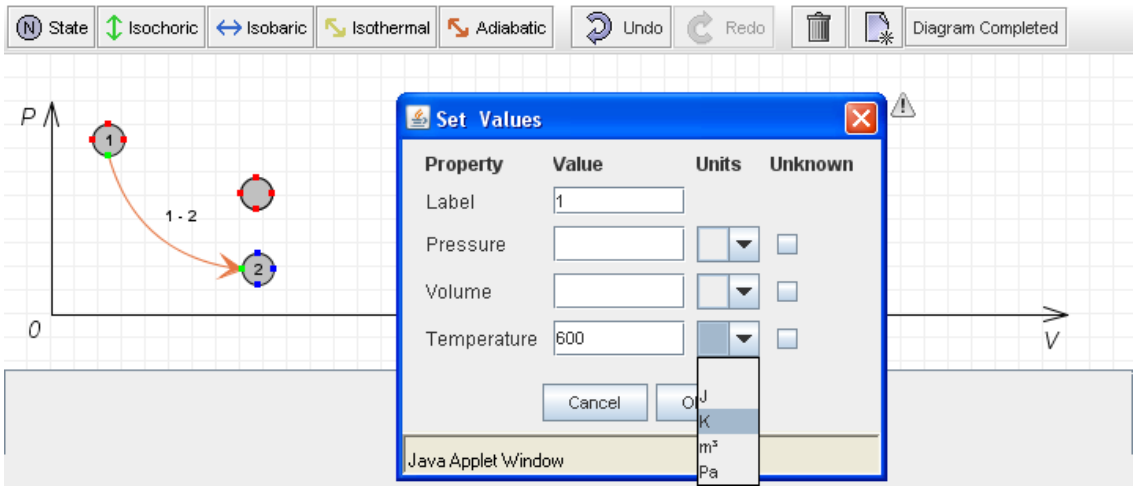


Figure 2.3: Adding a new state to the diagram

Specifying Some Constants

After the student has completed the diagram they must specify values for four constants which may be used in calculations in the following step. These constants are C_v (heat capacity at constant volume), C_p (heat capacity at constant pressure), R (ideal gas constant) and γ (used for adiabatic transitions). Figure 2.4 shows selecting a value for C_v by clicking the '...' button beside 'C_v' in the data summary. This step is included so that the data summary displays all information that the student could possibly need for

subsequent calculations in the next step.

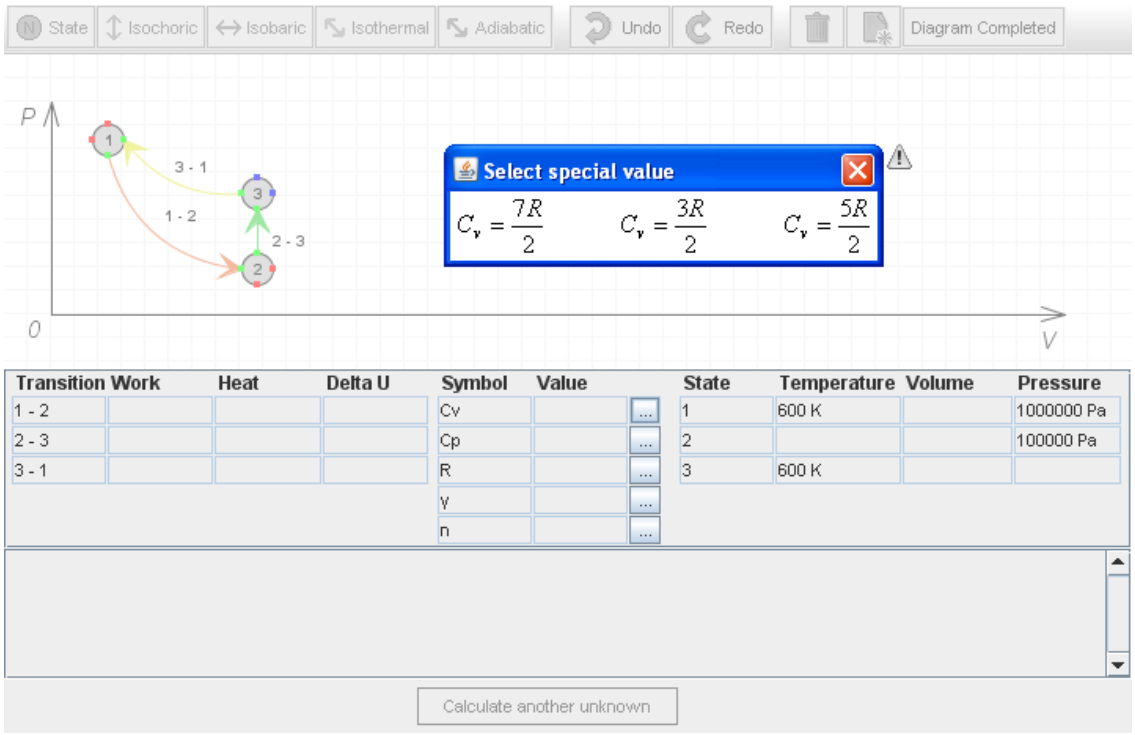


Figure 2.4: Selecting value for C_v

Calculating Unknowns

After the student has correctly selected values for the four constants they move on to calculating the unknown properties of states and transitions. Carrying out this step on paper involves using formulae to gradually build upon what is known about the cycle. The formulae used will depend upon what is known initially, and the type of transitions present in the cycle. Some of the formulae involve exponents and logarithms, which make it difficult to design an interface which is easy to use with a mouse and keyboard. Many people find entering mathematical expressions with a keyboard cumbersome, and for good reason. The method of entering expressions via a keyboard differs significantly from the usual method of using a calculator.

This stage raised the most significant issues of interface design. The task as I saw it was to find a compromise between two somewhat opposing requirements. The interface must firstly reflect how the problem would be solved on paper, and secondly be very quick and easy to use. The former is important to facilitate transfer, and the latter is important to minimize the cognitive load imposed by the interface so the student is able to concentrate on the problem solving task. While these two requirements are not entirely opposing, satisfying both to an appropriate level proved very difficult.

The steps required to find the value of a single unknown are:

1. Select the unknown to calculate
2. Select the formula group to use
3. Select the formula to use from this group
4. If necessary, rearrange formula by selecting from alternatives
5. Fill in all variables in the equation and specify the units of the result
6. Evaluate the equation to get the value for this unknown

The interface firstly requires the student to select which unknown they are going to calculate (Figure 2.5).

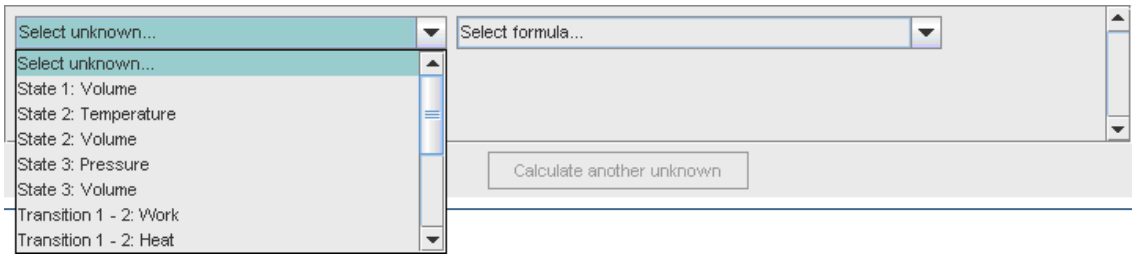


Figure 2.5: Selecting the unknown to calculate

Next the student must select the formula group they are going to use (Figure 2.6).

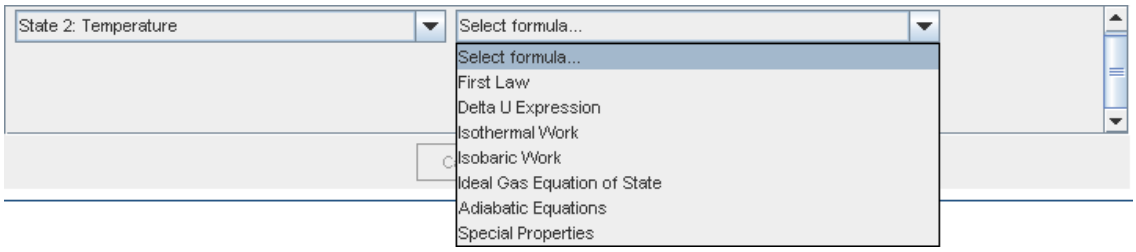


Figure 2.6: Selecting the formula group

After selecting the formula group, a dialog is raised displaying the formulae in that group (Figure 2.7). The student must select the correct formula from this dialog. Each group contains a number of formulae, some of which are valid under some circumstances, and some which are never valid.

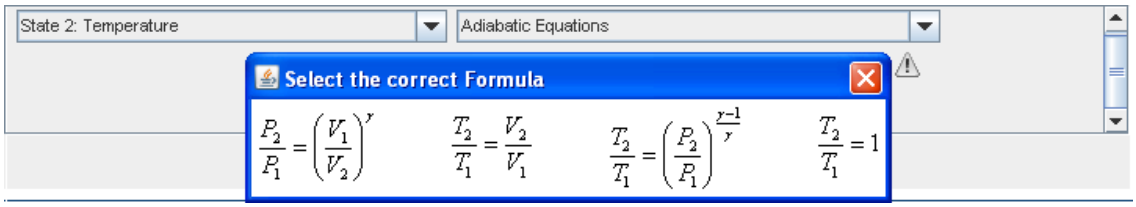


Figure 2.7: Selecting the formula from the group

Some formulae also need to be rearranged. Figure 2.8 shows the state of the interface after selecting a formula which needs to be rearranged. This step is skipped for formulae which do not need to be rearranged.

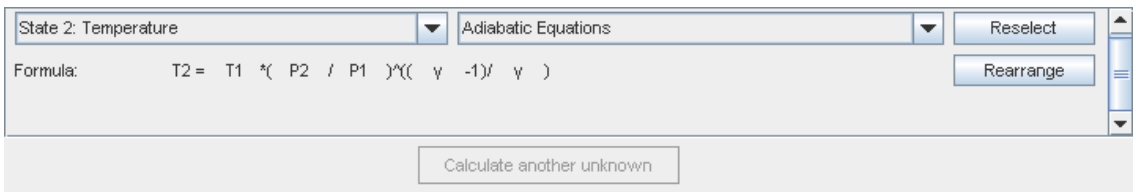


Figure 2.8: Formula needs to be rearranged

Figure 2.9 shows the dialog which is raised after clicking the Rearrange button. The student must simply make a selection from this dialog, exactly as in the previous step.

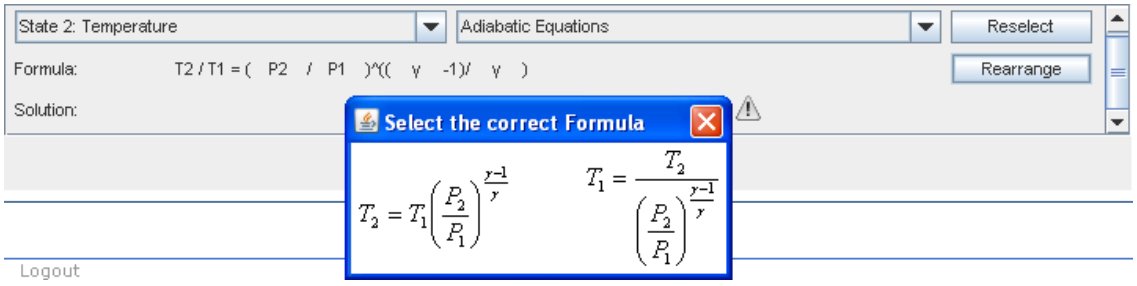


Figure 2.9: Selecting rearrangment

Now the formula is ready to be filled in (Figure 2.10). The student must provide values for all the variables on the right hand side of the equation, and select a unit for the result. The interface provides a textbox for each variable in the formula. The student must fill in each of these textboxes by selecting the value from the data summary. Values can be copied from the data summary into the currently selected textbox by double-clicking or right-clicking.

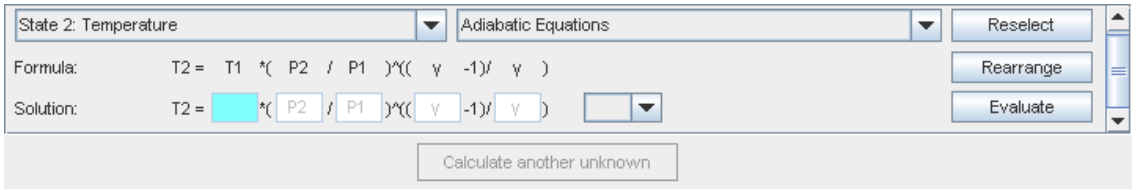


Figure 2.10: Filling in known values

2.2 Learning, Memory and Cognition

2.2.1 Generation Effect

The generation effect states that generating (creating) a stimulus item almost always results in that item being better remembered [25, 26] (see [27] for a review). It attempts to explain why and how forcing students to be actively involved in a learning task by requiring them to generate all or part of the material generally results in that material being better remembered. The generation effect can be described as "learning by doing". A large body of research has been focussed on this seemingly simple and intuitive concept. Despite this, some controversy still remains about many of the most fundamental particulars of the generation effect [27]. However, some fairly definite answers have been produced [27].

Early studies conducted to examine the generation effect involved the presentation of some type of paired-associates list to participants. Some common examples include word lists, number and letter bi-grams, mathematical equations and even nonsense word pairs [27]. In all cases half the pairs are provided to the participants, which the participant must simply read (e.g., `BRIGHT, DARK`). The remainder of the pairs are treated differently. The participants are given the first item of the pair (e.g., `BRIGHT, ____`) along with a rule that they must use to generate the second item of the pair (e.g., synonyms, antonyms, rhymes). Many other variations of this basic method were also common in early studies. These include reading or completing whole sentences [28], mathematical multiplication [29] or addition [30], and using anagrams with intact or jumbled solutions [31].

Several studies have examined the generation effect in the context of real learning scenarios such as arithmetic [29, 30] and even accounting [11]. However, the vast majority of study surrounding this effect has been performed in the context of rather artificial psychological analysis and experiments.

2.2.2 Cognitive Load Theory

Cognitive load theory (CLT) [32, 33] describes the operations of working memory in terms of its fundamental limitations. It observes that all but the simplest of cognitive activities appear to be subject to these limitations. CLT uses the concept of cognitive load to describe the limitations of our working memory. There are three basic types of cognitive load:

- **Intrinsic:** Results from the inherent complexity of the instructional material. This cannot be manipulated by instructional design. It can only be manipulated by altering the material itself, for example by splitting material so it can be presented as several separate parts to be recombined later.
- **Extraneous:** Results from the manner in which material is presented to students. This is under the control of the instructional designer because it is a direct consequence of instructional design. Choosing more appropriate methods of presenting material will reduce extraneous cognitive load which will generally increase the efficiency of instruction.
- **Germane:** The cognitive load devoted to processing, construction and automation of schemas. This is the load devoted to actual "learning" as it is defined by cognitive load theory. Effective instruction should promote germane cognitive load.

CLT describes knowledge in terms of schemas [32, 34]. Schemas are the fundamental construct for organising elements of information that are used together. A schema represents an abstract concept that can be applied in a range of contexts. For example, people will usually have an abstract idea of a tree in the form of a schema. This tree schema allows a person to effortlessly categorise and deal with the infinite complexity of trees. The abstract nature of these schemas also make them very suitable for dealing with problem solving [35]. They allow a person to deal with the infinite variety of a certain type of problem. The vast majority of learned, intellectual skills can be elegantly explained in terms of schemas.

CLT describes learning in terms of two critical mechanisms [32]; schema acquisition and transfer of learned procedures from controlled to automatic processing. It is often convenient to think of schemas being acquired in a dichotomous fashion; a person either has or has not acquired a schema. However, learning a new intellectual skill usually happens very gradually. When some new skill is first learned it is very difficult to apply it successfully. The application of this new skill does not occur in a fluent or effortless manner for some time. After some amount of time the application of a new skill switches from being controlled to automatic [32, 36, 37]. Controlled processing requires conscious attention and deliberate thought, automatic processing does not. Controlled and automatic processing are also conveniently thought of as dichotomous states. In reality, the transition from controlled to automatic processing usually happens gradually.

It is important to note that learning as described by CLT also imposes a cognitive load; germane cognitive load. Learning is in itself a cognitive activity, and so it is subject to all the same limitations of human cognitive architecture. Therefore, when designing instruction to achieve optimal learning it is essential to consider CLT. Instruction should generally be designed to limit the amount of unnecessary processing (extraneous cognitive load) and promote cognitive activity that is devoted to learning (germane cognitive load). However, it is often difficult to determine if some cognitive activity is extraneous or germane. It is often tied to the student's level of expertise. An activity which is germane for an expert is likely to be extraneous for a novice and vice versa. This is related to the degree of guidance provided which is discussed in the following section.

2.2.3 Guidance and Assistance Dilemma

The amount of guidance that a student should be provided when completing some learning task remains an open question. In fact the various teaching strategies that are advocated by experts can be distinguished by varying degrees and types of guidance. This spectrum of guidance ranges from unguided or minimally guided environments to direct instructional guidance. Advocates of unguided or minimally guided environments hypothesise that people learn best when they are required to discover or construct essential information for themselves [38, 39, 40]. Others argue that novice learners should be provided with direct instructional guidance on the required concepts and procedures [41, 42, 43]. Some examples of teaching strategies which are generally regarded to exhibit less than direct instructional guidance include discov-

ery learning, inquiry learning, experimental learning and constructivist learning. However, the degree of guidance provided can also be adjusted within each of these teaching strategies.

There are many ways to discuss guidance in education. One such way which appeared recently is known as the assistance dilemma [44]. The assistance dilemma has been explored in the context of ITSs [12]. Assistance is defined as any instructional affordance or change to the learning environment which makes the task easier (increases performance) or reduces mental effort (cognitive load). So in reality, assistance is more than just guidance, but the most common way of providing assistance is to increase guidance. The assistance dilemma poses the question: "How should learning environments balance information or assistance giving and withholding to achieve optimal student learning?" [12].

The matter of assistance remains a dilemma for two main reasons. Firstly, numerous experimental results seem to show both positive and negative effects from greater levels of guidance. Secondly, despite many advances, cognitive theory remains unable to predict when providing assistance will be beneficial or harmful.

Understanding the assistance dilemma requires understanding why the level of assistance is not correlated with learning outcomes [44]. High assistance during instruction can either be a "crutch" that harms learning or a "scaffold" that bootstraps learning. Consider the example of a child learning to tie their shoes. They need to be shown (provided a "scaffold") several times before they will be able to tie their own shoes. However, if their shoes continue to be tied for them (provided a "crutch") they will never learn how to do it themselves. Low assistance during instruction can either create a situation of desirable difficulty or undesirable difficulty [45]. For a novice, low assistance can create a situation of undesirable activity where their cognitive capacity is exhausted and they are forced to employ inefficient problem solving strategies [46]. However, low assistance can create a situation of desirable difficulty for experts where they are appropriately challenged and engaged (i.e., they are in their "zone of proximal development" [47]).

This seems to suggest that high assistance should be given to novices and assistance should decrease as expertise increases. This idea is backed up by many long standing notions such as the zone of proximal development [47]. This general rule has even been accepted to some degree among those advocating teaching strategies which generally provide lower levels of guidance. For example, pure discovery learning has been largely accepted as flawed, and other criticisms of "minimally guided" teaching strategies have been met with rebuttals claiming that these methods can and sometimes do support higher degrees of guidance. So again one might wonder what is the dilemma? We should just give novices high assistance and fade the assistance as expertise increases. However, the current theory does not provide guidelines on how much assistance should be provided initially, nor when or how quickly it should be faded. Nor can the current theory predict when an instructional demand (or cognitive activity) is desirable or undesirable (germane or extraneous). "The Assistance Dilemma remains unresolved because we do not have adequate cognitive theory to make a priori predictions about what forms and levels of assistance yield robust learning under what conditions" [44].

3 Design and Implementation

3.1 Goals and Hypotheses

3.1.1 Research Objectives

The primary objective of this research is to investigate what effect input method has on learning in an Intelligent Tutoring System (ITS). This investigation will be carried out by modifying Thermo-Tutor; an ITS designed to teach the fundamentals of solving problems involving thermodynamic cycles in closed systems. The modifications will include designing and implementing an experimental interface which requires the student to enter more information by typing as opposed to selection. The generation effect [26] suggests that this may be beneficial to learning. Other theory from psychology and cognitive science will also have to be considered during the design of this interface.

Thermo-Tutor is being developed in ASPIRE; a general purpose authoring system for developing Constraint-Based ITSs. A secondary aim of this research is to evaluate the effectiveness of ASPIRE as a general purpose authoring system. I wish to examine whether ASPIRE is capable of allowing a domain expert (e.g. a teacher) without programming expertise to create an ITS for their own domain.

3.1.2 Goals

In short, the goals of this research are:

1. Demonstrate the Generation Effect with an experimental interface for Thermo-Tutor
2. Evaluate the effectiveness of ASPIRE as a general purpose authoring system
3. Contribute to the set of available ITSs by completing Thermo-Tutor

3.1.3 Hypotheses

Correspondingly, my hypotheses are:

1. Requiring the student to provide more information by typing as opposed to selection will improve learning.
2. ASPIRE will prove to be an effective authoring system for authors with little or no programming expertise, but certain limitations will mean that some programming expertise is required for very complex domains.

The first hypothesis is supported by the Generation Effect [26] which states that self-generated items will, in general, be better remembered than items which are merely perceived by a person. There are other theories from cognitive science which seem to suggest the opposite may actually occur. This theory has been discussed further in Section 2.2

The second hypothesis is supported by previous evaluations of ASPIRE [17]. However, Thermo-Tutor is more complex than other ITSs which have been developed previously in ASPIRE. Because of this complexity, I believe that a significant amount of work which requires programming expertise will need to be carried out. It is currently unclear how much assistance ASPIRE will be in this case.

3.2 Completing Thermo-Tutor

Prior to this project the interface for Thermo-Tutor was considered complete. It was very similar to the interface described in Section 2.1.3. The few small changes that were made are described in Section 3.3. However, prior to this study, the constraint base of Thermo-Tutor was not complete. Development of constraints for the first step (drawing the diagram) was complete. Development of constraints for the second and third steps (specifying the constants and calculating unknown properties) had not yet begun. Significant effort during the first half of this project went into completing these constraints.

Generating constraints for the second and third steps of the tutor meant following ASPIRE's authoring process (described in Section 2.1.2). The most time consuming parts of this task include defining the ontology, entering ideal solutions and editing the constraints that ASPIRE generates.

Defining the ontology in ASPIRE involved considering all the problems and their solutions in order to devise an ontology capable of expressing the relevant concepts. It is important to get the ontology correct near the beginning of development because changes to the ontology often require subsequent steps in the authoring process to be repeated. It is also important that the ontology fully expresses the relevant concepts in the domain because this will have a great influence over the quality of the constraints that ASPIRE is able to generate.

Defining the ideal solutions was a particularly difficult task for the domain of thermodynamics for two main reasons. Firstly, there is no single order in which unknown properties should be calculated. Secondly, there is often multiple ways of calculating each unknown. ASPIRE's authoring process allows it to generate constraints for domains with multiple correct solutions (see Section 2.1.2). For such domains, the author would typically specify the all the correct solutions. ASPIRE would then analyse these solutions to generate constraints which would accept any of these solutions. Although the problems in Thermo-Tutor have multiple solutions, it would be very difficult to enumerate them in this fashion. It would be difficult to devise solutions which would match the student's solution as a whole. It would usually be possible for the student to produce a valid solution which takes parts from several of the correct solutions. The constraints that ASPIRE generates for domains with multiple correct solutions would not allow the student to use parts from several of the correct solutions.

I decided not to define the multiple ideal solutions in the usual fashion. I also decided not to define the order in which properties should be calculated. Instead, the ideal solution for calculating unknown properties is a set which enumerates all possible ways of calculating each unknown. Each calculation in the student's solution is examined in isolation and is considered correct if it matches one of the solutions in this set. One final constraint ensures all properties of all transitions are calculated before the solution is considered complete. The chosen representation of the ideal solution does not match the one assumed by ASPIRE. Due to this mismatch, many modifications were required to the constraints which ASPIRE generated (see Section 5.2). Thermo-Tutor is also not able to provide feedback which would help the student decide which property to calculate next.

After entering the ideal solutions for all problems, ASPIRE was able to generate the constraints for the second and third steps of Thermo-Tutor. The constraints which ASPIRE generated for the second step were sufficient. However, these were edited slightly to provide more concise feedback. These alterations involved making the constraints slightly more specific so that more specific feedback could be provided, and writing the feedback messages themselves.

As expected, the constraints that ASPIRE generated for the third step required substantial changes. These changes were required for two main reasons. Firstly, I had chosen a slightly unusual way of defining the ideal solution. This meant that many constraints had to be deleted for reasons described previously. Secondly, I wanted to provide more detailed feedback.

Several other changes were required because I wanted to provide more detailed feedback. These changes are described in more detail in Section 5.2. There are two ways I aimed to improve the feedback Thermo-Tutor provided while calculating unknown properties. Firstly, I wanted to provide some immediate feedback at certain stages. Secondly, I wanted the feedback to be able to provide the students with hints about what to do next.

I wanted Thermo-Tutor to provide immediate feedback at certain stages during the process of calculating an unknown property (described in Section 2.1.3). In particular, I wanted Thermo-Tutor to immediately inform the student when they made an error during each of the following substeps:

1. Select the formula group
2. Select the formula from this group
3. Rearrange the formula

These substeps are repeated for each calculation made during the third step in Thermo-Tutor. There are actually more substeps than in the process of calculating an unknown (see Section 2.1.3), but the student only receives feedback regarding these three substeps; so only these steps need to be considered when developing constraints. ASPIRE supports procedural problem domains with several steps and allows feedback to be provided separately for each step. In fact, the three main steps which make up Thermo-Tutor's problem solving procedure are defined in exactly this way. Usually, the author would define a step within ASPIRE for each point in the problem solving procedure where the answer must be checked, and feedback should be provided. It would be possible to define a step for each of the three substeps described above. However, ASPIRE requires an ideal solution to be defined for each step in the problem solving procedure. This would require several parts of the ideal solution to be duplicated, which would make these ideal solutions more difficult to create and maintain.

Instead I altered the constraints to allow feedback to be provided for each of the substeps above. The applet keeps track of which step (or substep) the student is currently working on. This information is sent with every submission of the student's solution. The relevance condition of all constraints includes a check of the current step, and only those constraints intended for the current step will be relevant. The constraints which included these substeps are very similar to the constraints which ASPIRE would have generated if I had chosen to define the substeps as real steps within ASPIRE. The difference is that they all operate on a single ideal solution, which makes this ideal solution easier to create and maintain.

Other changes were required to the constraints because I wanted the feedback to provide some hints to the student regarding what to do next. For example, when a student selects an incorrect formula group they will be informed of this with a simple message. If the student requests more detailed feedback, they will be told which formula groups they could use. ASPIRE did not generate these constraints. However, writing them was fairly easy because ASPIRE generated constraints which I could base my own on.

Finally, the feedback messages corresponding to each constraint had to be completed. The feedback message for a constraint is shown to the student when the constraint is violated. It is essential that these messages are well designed because this is the main form of assistance that the tutor provides to the student. These messages should contain the information the student requires to correct their misunderstanding which lead to the error. ASPIRE automatically creates generic feedback messages, however these are intended to be modified to provide more meaningful feedback.

3.3 Control Interface

A few small changes were made to the original interface of Thermo-Tutor to create the control interface for this study. The most significant change was to disallow typing when making calculations. The original interface gave the student the option of either typing values or selecting them from the data summary when performing some calculation. Typing was disabled in the control interface so the two modes of interaction were cleanly separated between the control and experimental interfaces. The student would be forced to select in the control interface, and type in the experimental interface.

Several other minor changes were also made to the original interface. These changes were fairly small improvements to make the interface easier to use. None of these changes would have a fundamental affect on the users' experience with Thermo-Tutor from a pedagogical point of view.

3.4 Experimental Interface

The experimental interface differs only in the final step of the problem solving procedure; calculating unknowns. There are two main reasons why this area of the interface was chosen for redesign. Firstly, the design of this part of the interface raised the most questions in the beginning. Secondly, calculating unknowns is generally the most complicated, challenging and time consuming part of solving these types of problems. It is expected that students will spend the majority of their time calculating unknowns, so this is the area of the interface which is most likely to benefit from redesign.

The experimental interface requires the student to do a lot more typing. It is expected that this will have a negative effect on the students' problem solving efficiency; however this is not the primary requirement of the interface. The primary goal of Thermo-Tutor is to teach problem solving of thermodynamic cycles in closed systems. Therefore, the interface of Thermo-Tutor should be designed in terms of pedagogical factors, rather than efficiency or other factors from Human Computer Interaction.

3.4.1 Calculating Unknowns

In this section I will describe how a student calculates an unknown in the experimental interface. In this new interface, the steps required to calculate a single unknown are:

1. Select the unknown to calculate
2. Select the formula group to use
3. Type the formula in the form which it will be used
4. Type the equation which uses this formula and specify the units of the result
5. Evaluate the equation to get the value for this unknown

There are a few very important differences to note. The first two steps are identical. The third step differs in two ways. Firstly, the formula is manually typed in rather than selected from a set of alternatives. Secondly, the formula is always entered in the form that it will be used, as opposed to specifying the "standard" version and then specifying the "rearranged" version. This change makes the previous rearrangement step redundant. The fourth step corresponds to the fifth step in the control interface. This step requires the student to use this formula, which involves entering it again with variables replaced by numeric values. Finally the student can evaluate the equation to obtain the result for subsequent calculations exactly as in the control interface.

The interface firstly requires the student to select which unknown they are going to calculate exactly as in the control interface (Figure 3.1).

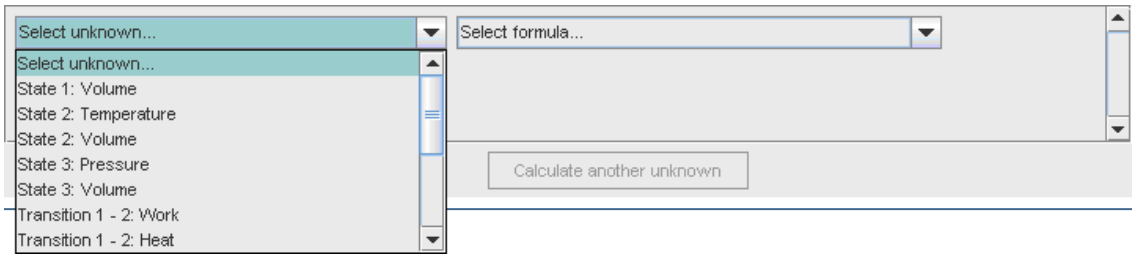


Figure 3.1: Selecting the unknown to calculate

Next the student must select the formula group they are going to use exactly as in the control interface (Figure 3.2).

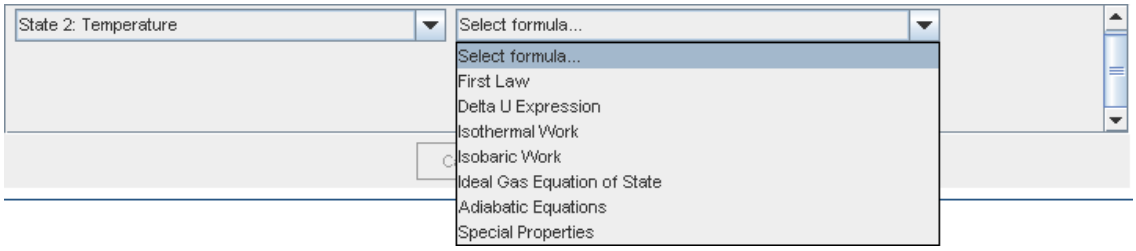


Figure 3.2: Selecting the formula group

After selecting the formula group, the student must type the correct formula in the textbox (Figure 3.3). This formula must be entered in the form that it will be used; there is no additional step for rearranging the formula.

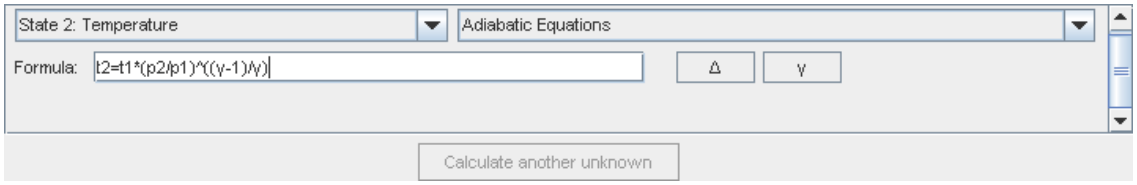


Figure 3.3: Entering the formula from the group

If the student struggles to enter the correct formula a dialog is raised to help them (Figure 3.4). This dialog is raised if the student enters three incorrect, unique and syntactically valid formulae. This dialog shows the formulae in the group which the student previously selected. It shows these formulae in their "standard" form; the student will often be required to alter them slightly. Often the formula will have to be rearranged to have the correct subject, and some of the terms may need to be altered to refer to the correct properties. For example, the formulae in Figure 3.4 refer to T_1 and T_2. These may need to be altered to instead refer to T_2 and T_3.

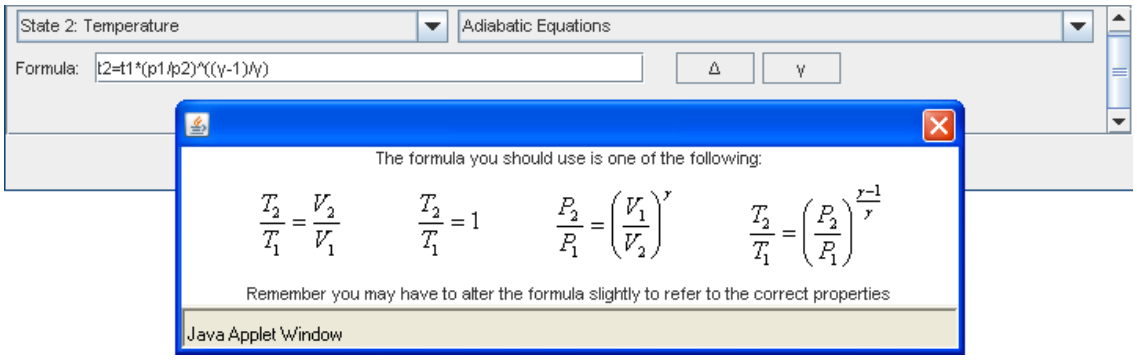


Figure 3.4: Assistance with entering the formula

When the student has entered the correct formula, it must then be filled in with values from the data summary (Figure 3.5). The student must enter the entire formula again with variables replaced by values and select a unit for the result.

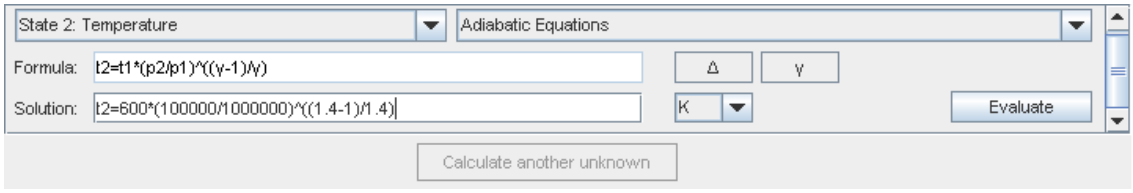


Figure 3.5: Filling in known values

3.5 Motivation for Redesign

To understand the tradeoffs between the control and experimental interfaces it is necessary to understand how these two designs came about. The control version was completed as part of a summer project I conducted to continue development of Thermo-Tutor [24]. At the time of designing the interface I recognised

that there were several factors influencing the design and that it would not be possible to satisfy them all. I opted for a design which would allow the student to solve problems more efficiently.

An earlier design for the interface was very similar to the experimental interface. It was recognised that this interface was likely to result in better transfer because it more closely reflected the process of using a calculator. However, it was thought that this extra work would be unnecessary and tedious for the intended user of Thermo-Tutor. Thermo-Tutor is intended for 200 level engineering students. These students will be very familiar with using a calculator. The task of entering these expressions will be relatively trivial compared to considering what is known and determining how to best use the formulae. There is some truth in my previous observations, however I believe I underestimated the value of what I considered to be "unnecessary and tedious" work.

I justified this decision in terms of cognitive load by assuming that a more efficient interface will reduce the extraneous cognitive load imposed on the student which will allow the student to devote more of their effort to actual learning. This assumption is somewhat justifiable from the point of view of Cognitive Load Theory (CLT) [32]. However, the effect that this strategy has on learning remains unclear.

According to CLT, the underlying problem with this strategy is it relies on the student behaving in a particular way. A better strategy would attempt to encourage the student to behave in a way which is beneficial to learning, as opposed to simply hoping that they will behave in such a way. This can be described in terms of cognitive load theory. The original strategy aims to reduce extraneous cognitive load in the hope that the student will devote more attention to the material (increasing germane cognitive load). A better strategy would aim to reduce extraneous cognitive load, but it would also aim to actively promote germane load by forcing the student to engage in cognitive activity that is beneficial to learning.

There are several ways of examining the differences between the control and experimental interfaces. Perhaps the simplest is from the point of view of interaction style. The control interface takes input via selection; the experimental interface takes input via typing. This primary aim of this research is to investigate what effect these two interaction styles have on learning.

3.6 Design Tradeoffs

When seeking research theory to help guide the design I discovered a large amount of material from cognitive science and psychology that seemed relevant. The three main ideas that I had to consider were the generation effect, cognitive load theory and guidance. The generation effect states that items which are generated are generally better remembered than items which are merely read or perceived. Cognitive load theory distinguishes between different types of cognitive load which is helpful when discussing the benefits of some cognitive activity. Guidance is about determining how much assistance should be provided to the student, and how much work they should be required to do themselves.

The generation effect predicts that material which the student has some part in generating or creating will generally be better remembered. This would suggest that requiring the student to type formulae as opposed to just selecting them from alternatives will result in those formulae being better remembered. Memorising formulae is generally not considered essential, however memorising them is one step closer to understanding them. A formula exists to represent some meaningful relationship between a set of properties. Understanding these relationships between properties is fundamental to thermodynamics. By forcing the student to type these formulae it seems likely that the student will begin to recognise the relationships that these formulae represent. This should eventually enable the student to better understand these relationships. I was concerned that the control interface would allow the student to select and fill in a formula without having to examine the structure or meaning of that formula closely.

Cognitive load theory distinguishes between three main forms of cognitive load; intrinsic, extraneous and germane. As instructional designers, we are primarily concerned with extraneous and germane cognitive load. We generally aim to reduce extraneous and promote germane cognitive load. The control interface aims to reduce extraneous cognitive load by providing the student with a very efficient interface, however it fails to promote germane cognitive load because it does not require the student to engage with the material. The experimental interface also aims to reduce extraneous cognitive load by providing a certain degree of leniency, and it aims to promote germane cognitive load by forcing the student to focus on the meaningful aspects of the material. It is hoped that this will benefit learning because the student will engage more deeply with the material.

The issue of guidance is at the heart of much debate in the field of education. There are advocates for all levels across the spectrum from minimal to full guidance. There are many strong arguments for both sides of this debate. However, it is generally agreed that novices require very high guidance, and as expertise increases the level of guidance required decreases. An ideal system would be aware of the level of expertise of the student, and use this to provide the appropriate level of guidance. The experimental interface provides less guidance than the control interface. In the control interface the student is provided with a set of alternative formulae to choose from. On the other hand, the experimental interface allows the student to enter any formula they wish. This formula need not even be syntactically valid. When the student makes an error in the control interface, they can reselect from the same set of alternatives. They could try all alternatives until they select the correct one by chance. In the experimental interface, it is very unlikely that a student will be able to enter the correct formula by chance.

We attempted to increase the level of guidance provided to novices by the experimental interface. When a student enters three incorrect, unique, syntactically valid formulae we assume this student must be a novice. Consequently, a dialog is raised which displays a set of alternative formulae. This does not provide the same level of guidance as the control interface because the student cannot select from these alternatives. The alternatives are also shown in their "standard" form; often the student will be required to adapt them slightly to suit their purpose. This usually involves rearranging the formula and adjusting the terms slightly to refer to the correct properties. It is not possible for the two interfaces to provide the same level of guidance because requiring the student to generate more is inherently guiding them less.

3.7 Added Complications

The experimental interface is far less restrictive in that it makes more of the task of constructing the solution the students' responsibility. This allows the student to enter a much wider range of possible solutions. This means there are more ways in which the student can be incorrect which places greater demands on Thermo-Tutor.

Thermo-Tutor must be capable of catching, analysing and providing feedback relating to these incorrect solutions. To support this, Thermo-Tutor has to be capable of parsing mathematical expressions to get a result, check for syntax errors and deal with equivalent expressions. Thermo-Tutor also had to handle some additional errors and achieve an appropriate degree of leniency so that the student would not be unnecessarily halted by meaningless errors.

Thermo-Tutor had to handle mathematical expressions in a more sophisticated way. Originally mathematical expressions only had to be evaluated to get the result so it could be used for subsequent calculations. These expressions could be guaranteed to be one of a small set that were provided. The experimental interface allows the student to enter arbitrary formulae and expressions. The student could enter expressions which are different but equivalent to the ideal solution, or contain syntax errors. Both these cases had to be handled appropriately.

Solutions which are different but equivalent to the ideal solution should be accepted as correct. To partially support this, a number of common equivalent solutions were defined in the ideal solution so these would be accepted as correct. There are still many more equivalent solutions which are not included. To allow the student to include redundant parentheses, I also implemented a simple parser which removed these from the students' solution before it is compared against the ideal solution. A better solution might be to define the ideal solution using prefix notation. The student would continue to provide the solution in infix notation, so these solutions would have to be converted to prefix notation before they are checked against the ideal solution. This solution would work well because formulae and expressions defined in prefix notation can be parsed without ambiguity and do not require any parentheses. This would eliminate some of the ambiguity in the solution and make it easier to define the remainder of the acceptable equivalent solutions.

The student could also enter solutions which contain syntax errors. These had to be caught by the parser so that the student could be informed that they have made a syntax error.

The process that the experimental interface requires the student to follow is much less restrictive. This meant that the student can make a number of errors which are simply not possible in the control interface. For example, in the control interface the student selects a formula group and then they can only select a formula from that group. In the experimental interface the student selects a formula group and is the free

to enter any formula they wish. The student could select one valid formula group, and then enter a valid formula from another group. Additional constraints had to be written to catch this and other errors.

The student can also enter expressions which are incorrect in some meaningless way. In these cases it would not be appropriate to consider the students' solution incorrect because the student would be forced to spend time correcting this error with little or no educational benefit. Only errors which represent a genuine misconception should be considered incorrect because forcing the student to correct these errors will be beneficial to their learning. There are several ways the experimental version of Thermo-Tutor provides a degree of leniency to support this goal. For example, formulae and expressions are not case sensitive and they can include redundant parentheses. Students are also able to enter some formulae and expressions which are not identical but equivalent to the ideal solution.

3.8 Tools Used

Thermo-Tutor was developed using ASPIRE. ASPIRE is written in Allegro Common Lisp and runs on the AllegroServe web server on Microsoft Windows XP. ASPIRE also makes extensive use of JavaScript for the web interfaces it provides, and XML for transmission and temporary storage of data. ASPIRE uses the AllegroCache Object Oriented database for storing the majority of its operational data. The interface of Thermo-Tutor is provided by a Java applet, and communicates with ASPIRE via XML over HTTP.

Development for this research involved several different of technologies. The majority of the development was focussed on the interface of Thermo-Tutor which is developed in Java. Some JavaScript was also required to customise the parts of the interface provided by ASPIRE. The solution format of the experimental version was also altered, which required changes to the Java code which transmits XML to ASPIRE. Constraints are written in a purpose built language which is a subset of Common Lisp.

4 Evaluation

4.1 Description

An evaluation study was conducted at the University of Canterbury as part of a 200-level course covering transfer operations and thermodynamics. Students in this course are taught the relevant theory in weekly lectures and are given the opportunity to practice their skills in tutorials and homework exercises.

4.2 Participants

Students from ENCH 292[1] voluntarily participated in the evaluation study. Of the 63 students enrolled in the course, 32 completed the pre-test and 21 logged into Thermo-Tutor.

4.3 Evaluation Plan

An evaluation was planned to take place in two one hour sessions on the 11th and 18th of September. Students would be divided among the two sessions according to the first letter of their surname; A–L would attend the first session and M–Z would attend the second. Each student would attend only one session and would be randomly assigned to either the control or experimental group. Each student would be given instructions to explain how to log into ASPIRE and assigned them to either the control or experimental group. A between-subjects evaluation design would be used; each student would remain in the group they were assigned to for the entire evaluation. The control and experimental groups would use the control and experimental interfaces respectively.

All students would complete a pre-test (see Appendix A.1) in the lecture on the 11th of November, and a post-test (see Appendix A.2) immediately after the one hour evaluation session which they attend. Both the pre-test and post-test would be administered on paper and would be of comparable complexity. Both tests would be designed to evaluate understanding of the problem solving procedure and the formulae. Students would also be required to complete a questionnaire (see Appendix A.4) as part of the post-test which would ask each student to rank the system across five aspects, and also request comments and suggestions for future improvement.

4.4 Revised Evaluation Plan

When the results of the pre-test were examined it became apparent that we would need to make changes to the evaluation plan to obtain more meaningful results. The students achieved more highly than was expected (see Table 4.1). The students had previously completed ENCH 291[2], which also covered topics relevant to Thermo-Tutor. We originally considered performing the evaluation as part of ENCH 291, however this course occurred in the first semester (which would not leave sufficient time for development), and the lecturer believed the level of expertise of the students in ENCH 292 would be more appropriate for Thermo-Tutor.

The pretest included three questions (see Appendix A.1) for a maximum possible mark of 4. Students could earn one mark for questions 1 and 2, and two marks for question 3. Half marks were also given for small errors. Of the 32 students who completed the pre-test, just one student failed to solve question 1, and 6 students made a partial error in question 3. However, 12 students made an error in question 2. The

[1] ENCH 292 information available at http://www.canterbury.ac.nz/courseinfo/GetCourseDetails.aspx?course=ENCH292
[2] ENCH 291 information available at http://www.canterbury.ac.nz/courseinfo/GetCourseDetails.aspx?course=ENCH291

error rates for questions 1, 2 and 3 were 3%, 36% and 19% respectively. Due to the ceiling effect it seemed unlikely that we would be able to measure a statistically significant difference between the problem solving ability of the control and experimental group using a post-test similar to this pre-test. This problem was not detected in time to change the post-test for the first evaluation session, however we decided to change the post-test for the second evaluation session.

It was decided that the post-test should be changed to only examine the students' knowledge and understanding of the formulae. This decision was made on the basis of the results from the pre-test. Although the vast majority of students were extremely competent at solving these types of problems when they were given the formulae, 36% were unable to reproduce the formulae, even when provided with alternatives in multi-choice questions. It was decided that a new post-test (see Appendix A.3) should be devised which would examine the students' knowledge and understanding of the formulae.

4.5 Technical Difficulties

In addition to the ceiling effect that required the evaluation plan to be modified, technical difficulties were experienced in both evaluation sessions. In the first evaluation session many students experienced an error which caused the Thermo-Tutor applet to cease responding. This error was caused by students clicking the "Continue" button on the webpage provided by ASPIRE. This button was not being used by Thermo-Tutor and consequently had not been tested. Unfortunately several curious students were clicking this button which caused the applet to wait indefinitely for a response from ASPIRE which would never arrive. Of the 17 students who participated in the first evaluation session, 10 experienced this error at least once. There was also another issue which caused some images in the applet to not render. This issue was related to the environment in the lab where the evaluation was conducted; namely the specific version of Java and Internet Explorer. Thermo-Tutor had not been thoroughly tested in this environment. These images displayed units, constants and formulae (see Figures 2.3, 2.4 and 2.7). The fact that these images were not always visible would negatively influence the usability of the interface. However, neither of these errors prevented any student from being able to use Thermo-Tutor. These issues were corrected for the second evaluation session.

Unfortunately the second evaluation session saw more severe technical issues. An error which appeared to originate in ASPIRE itself caused the server to crash. The error caused a thread to exceed its system memory limit. Allegro handled this error by locking the offending thread and spawning another. This repeated several times before Allegro had consumed a large amount of system memory and ceased to respond. This occured approximately 5 to 10 minutes into the second evaluation session. It took approximately 40 minutes to diagnose the issue and restart the web server. When the web server was finally back online there was only approximately 5 minutes remaining in the evaluation session. It was decided that the students had not used Thermo-Tutor for long enough to obtain meaningful results from either the post-test or questionnaire. Consequently this data has been excluded from the analysis.

4.6 Results

Unfortunately the results which could be obtained from the evaluation were severely limited due to the ceiling effect and several technical difficulties. The most useful results were obtained during the first evaluation session. Log files were analysed to obtain some general statistics regarding the usage of the tutor. The results from the Likert scales in the questionnaire were statistically analysed and the students' comments were carefully interpreted. Finally, the student models were analysed to obtain learning curves which show the rate at which students were learning. This section presents the results of this analysis.

4.6.1 Student Performance

Before examining other results it is useful to look at some general statistics regarding student performance in the pre-test and in Thermo-Tutor. These factors are important because they could influence the interpretation of the results in Sections 4.6.2 and 4.6.3. Table 4.1 shows some of these general performance statistics along with scores from a T-Test to look for differences between the control and experimental groups. It shows the mean and standard deviation for the control and experimental group for several factors. These results came from analysing the log files for the 17 students who took part in the first evaluation

session. There were 7 students in the control group and 10 students in the experimental group. In order from top to bottom these factors are; pre-test score, number of submissions made, number of successful calculations performed, number of problems attempted, number of problems completed, number of problems completed up to the final step (performing calculations) and the session length (in minutes).

	Control	Experimental	p
Pre-test Score (%)	77.5 (23.8)	87.5 (15.0)	0.41
Submissions	75.9 (50.2)	53.1 (35.8)	0.33
Successful Calculations	12.3 (13.3)	3.0 (5.0)	0.12
Problems Attempted	1.7 (0.48)	3.2 (2.2)	0.07
Problems Completed	0.7 (0.95)	0.1 (0.32)	0.14
Problems Completed up to Final Step	1.0 (0.82)	0.4 (0.52)	0.12
Session Length (mins)	94.6 (17.6)	80.4 (28.6)	0.23

Table 4.1: Mean student performance statistics (standard deviations are shown in parentheses)

It is worth noting that there was no significant difference between the pre-test scores of the control and experimental groups. This means that these two groups belong to the same population; so it is valid to make direct comparisons between them.

It appears that the control group exceeded the experimental group in most areas. They seemed to have solved more problems, successfully completed more calculations, completed more problems (both in whole and up to the final step), and spent longer using the tutor. While none of these factors are significantly different.

Interestingly, it appears the experimental group attempted more problems; and this difference is marginally significant. This may have been due to the added difficulty of using the experimental interface. Perhaps students found the experimental interface so difficult to use that they simply moved on to the next problem, and did not try the final step for any of the subsequent problems they attempted. Some of the technical difficulties such as images not rendering also seemed to affect the experimental interface more often. This may have caused students to simply move onto the next problem after the first or second step. There was evidently some confusion about how to use Thermo-Tutor; one student in particular solved only the first step of every problem.

4.6.2 Questionnaire

Some of the participants were given a questionnaire intended to allow them to express their subjective opinions of Thermo-Tutor. 10 students completed this questionnaire, 4 from the control group and 6 from the experimental group. Table 4.2 shows the mean results from this questionnaire for both the control and experimental groups (standard deviations are shown in parentheses). The first two items in this table were asked as questions, the student was asked to give a ranking for the final three factors (see Appendix A.4). For each question and ranking students were asked to select a value on a 5 point Likert scale, with one representing poor/low, and five representing excellent/high.

Question	Control	Experimental	U	p
Overall quality of Thermo-Tutor	3.50 (1.05)	3.63 (0.48)	11	0.41
Quality of feedback provided by Thermo-Tutor	3.25 (1.08)	2.75 (0.96)	6	0.08
Mental effort required	3.67 (0.52)	3.00 (0.82)	8.5	0.76
Problem difficulty	3.17 (0.75)	2.75 (0.96)	12	0.5
Enjoyment	3.25 (0.76)	3.25 (1.50)	8.5	0.22

Table 4.2: Mean Likert scores for Thermo-Tutor and Mann-Whitney U Test results (standard deviations are shown in parentheses)

Table 4.2 also shows the results from performing a Mann-Whitney U Test for each question. For all results, there were 4 students in the control group and 6 in the experimental group ($N_1 = 4$ and $N_2 = 6$). None of the differences are significant at the p=0.05 level. However, there is a marginally significant difference between the responses from the control and experimental group for the second question (p=0.08). It appears that the experimental group found the quality of the feedback they received to be lower. This is an interesting result because the level of feedback did not differ greatly between the control and experimental versions of Thermo-Tutor. The experimental version was actually capable of providing slightly more feedback regarding errors that were not possible in the control version. These errors are possible in the experimental version because of the added input flexibility it provides (see Section 3.4). However, it appears that the students who used the experimental version felt they still needed more help.

These statistics should be interpreted with caution. It is important to remember that only 10 students in total completed the questionnaire. I am only willing to draw the conclusion I mentioned above because it corresponds with the observations I made during the evaluation period. These statistics should not be used to draw definite conclusions.

These results vary greatly, but seem to be somewhat positive. The students were also asked three open-ended questions to encourage them to share their thoughts about the strengths and weaknesses of Thermo-Tutor and what they thought could be improved. Some general thoughts, comments and suggestions were expressed in the answers to these questions:

- Several students found the feedback helpful. One student liked that Thermo-Tutor informed him of errors in his diagram. Another found the feedback was simple and direct for small errors.
- Several students in the control group found it useful to have all the formulae they needed given to them.
- Several students found it enjoyable and easy to draw diagrams in Thermo-Tutor
- Several students enjoyed solving these problems on a computer because of added interactivity, ease of use and for the simple fact that they enjoy using a computer.
- Several students found the feedback was not very helpful. This made it difficult for them to locate and correct their errors. One student suggested that Thermo-Tutor should indicate the order in which variables should be calculated.
- Several students indicated that there is a need for more help with using the tutor. One student suggested there should be a brief online tutorial.
- One student in the experimental group found it difficult to type formulae and expressions. They suggested that the tutor should help with syntax.

4.6.3 Learning Curves

Learning curves allow us to analyse whether some measurable objects are being learnt [48]. We can use learning curves to measure the rate at which constraints are being learnt. Learning curves plot the number of times the object (constraint) is relevant against the proportion of times it was incorrectly used (violated). If the object in question is being learnt, we should be able to obtain a good fit to an exponential curve.

Due to technical difficulties, only the results from the first evaluation session can be analysed. These results came from students using the system for approximately an hour and a half on average. It would usually take longer than this to gather sufficient data to plot accurate learning curves. Therefore, the results of this analysis should be interpreted with caution. Figures 4.1 and 4.2 show the learning curve obtained for the control and experimental group respectively.

Both the control and experimental groups show a moderately good fit to an exponential curve (both R^2=0.74). The learning rates of the control and experimental group appear to be nearly identical (as shown by the slope). The height of the curves seem to suggest that the experimental group made slightly more errors throughout the evaluation period. However, this difference is not significant, and the data points are very similar.

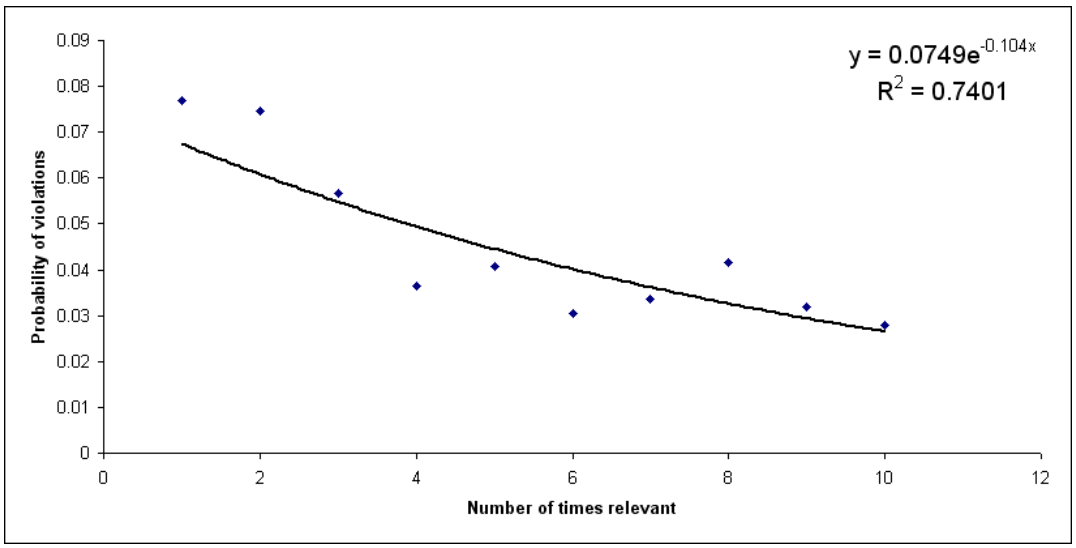


Figure 4.1: Learning curve for the control group

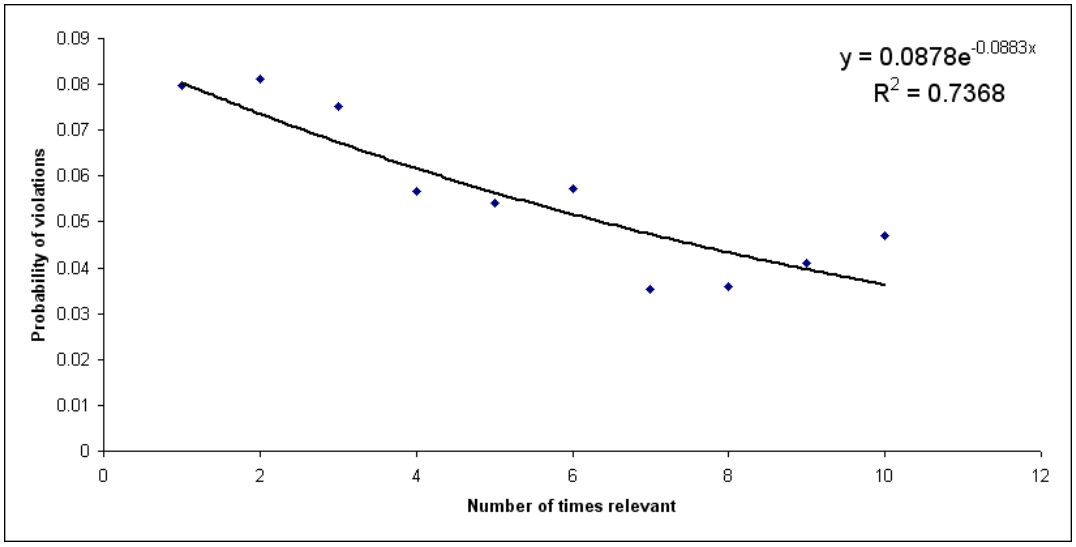


Figure 4.2: Learning curve for the experimental group

5 Discussion and Future Work

5.1 Possible Improvements

From the comments expressed in the questionnaire, and my personal observations of students during the evaluation and their log files, there are a number of things which require improvement.

Provide more detailed feedback

Some students found the feedback that Thermo-Tutor provided while making calculations was insufficient. This is primarily because the set of constraints that describe this part of the problem solving procedure are still quite simple. For this final step Thermo-Tutor makes no attempt to model the overall process; it only models each calculation in isolation. So it is not able to help the student decide which property to calculate next. Each calculation is also modelled in terms of formulae and expressions, as opposed to variables and values. So Thermo-Tutor is not able to provide detailed feedback regarding what part of the calculation is wrong. It is only able to inform the student that one or more of the formula, expression, result or unit is incorrect.

Provide more assistance to novices

It seems from the statistical analysis of the Likert scale results that the students who used the experimental interface felt they needed more help. Providing more detailed feedback, as described above, would help alleviate this problem to some degree. Another way to alleviate this problem would be to provide more assistance to novices. This would require two challenges to be overcome. Firstly, we need some way of classifying the expertise of a student. ASPIRE maintains a student model that would help with this. Secondly, we need to decide how much assistance to provide given some level of expertise. One possible way may be to allow "novices" to use the control interface, and require "experts" to use the experimental interface. This could occur on a per-problem basis by examining the student's expertise and the problem's difficulty. Alternative, the interface could adapt more dynamically. For example, after a certain number of incorrect attempts we may allow the student to revert to the control interface (based on the assumption that they are a "novice").

Accept a wider range of solutions

Some students in the experimental group found that Thermo-Tutor would not accept their perfectly valid solution for calculating some property. This was due to students using solutions which Thermo-Tutor simply did not accept. Thermodynamics is a vast topic; there are often many different ways of performing some calculation. Thermo-Tutor was designed to accept only solutions which students were likely to produce given what they were taught in the course. However, some students used valid problem solving techniques that were not considered during development of Thermo-Tutor. There may be some solutions which should be discouraged for pedagogical reasons, however it is unlikely that these techniques should be simply regarded as incorrect in the same manner as a truly incorrect solution. This is a greater issue in the experimental version, because there are many valid formulae that are not supported by the system. If the student enters one of these formulae in the experimental version, they are simply informed that they are incorrect. It is not clear to the student whether this means their solution is incorrect, or simply not

supported by the system. In the control version, these formulae simply cannot be selected; so it is clear to the student what formulae are valid.

Provide increased leniency

Some students in the experimental group found that Thermo-Tutor was very particular about the format it required when calculating some property. These students would perform a calculation which was semantically identical to one which Thermo-Tutor was aware of. However, because of some trivial syntactic difference, Thermo-Tutor would not accept this solution. Some work has been done to allow Thermo-Tutor to be lenient to some extent. Solutions with any amount of redundant parentheses will be accepted, and many equivalent ways of expressing solutions will be accepted. However, there are many more equivalent ways of expressing mathematical expressions which will not be accepted as correct. Detecting that two mathematical expressions are equivalent is non-trivial in general.

Provide a wider variety of problems

Several students became bored with the type of problems that Thermo-Tutor currently supports. This is likely to be due to the skill level of the students that took part in the evaluation. The lecturer and tutor were surprised at the level of competency the students demonstrated in these types of problems. It is believed that Thermo-Tutor may be more appropriate for first year students. However, it would be good to support a wider range of problems in Thermo-Tutor. Any significant increase in the complexity of the problems supported by Thermo-Tutor is likely to require alterations to the ontology and constraint base. This would require a significant amount of work, but would increase the value of Thermo-Tutor greatly.

5.2 ASPIRE

The secondary goal of this research was to evaluate the usefulness of ASPIRE as a general purpose authoring system for constraint-based ITSs. The primary advantage of using ASPIRE is that it will generate the domain model. Therefore to evaluate ASPIRE's overall usefulness we should examine its ability to generate this domain model. The domain model of constraint-based ITSs is modelled by a set of constraints which describe the domain concepts. ASPIRE's most difficult and important task is to generate this set of constraints. In order to evaluate the quality of the domain model ASPIRE generates, we should examine the quality of these constraints.

5.2.1 Generating the Domain Model

For the control version of Thermo-Tutor, ASPIRE was able to generate all the necessary syntax constraints. However, some syntax constraints had to be discarded. Of the 148 syntax constraints that ASPIRE generated, 8 were discarded leaving a total of 140 syntax constraints in the control version of Thermo-Tutor. These syntax constraints were invalid because of the substeps I included to make the definition of the ideal solution more convenient (see Section 3.2). These constraints could have been adapted to work correctly for the substeps, however they were not necessary because the errors which they were designed to catch were not possible with the interface of Thermo-Tutor. Several other syntax constraints which were necessary were adapted to work correctly with the substeps. Finally, all the feedback messages were rewritten, as is intended in ASPIRE's authoring process.

ASPIRE was also able to generate a sufficient set of semantic constraints for the first and second steps in Thermo-Tutor. The semantic constraints for the third step of Thermo-Tutor required the most substantial changes because of the way I chose to define the ideal solution. However, all the semantic constraints underwent significant changes, including those for the first and second steps, in order to provide more specific feedback.

ASPIRE generated 142 semantic constraints. Of these 45 were removed because they were unsuitable for the way I chose to define the ideal solution, and 32 were removed because they were overly specific. These overly specific constraints were the result of insufficient data (ideal solutions) to allow the constraint-induction algorithm to fully generalise. This left a set of 65 semantic constraints, which underwent significant changes to make the 93 semantic constraints in the final version of Thermo-Tutor. The

changes that were made to the original 65 constraints varied greatly, and the process was by no means straight forward. Consequently, it is difficult to succinctly describe these changes. I will attempt to give an overview of the more significant types of changes that were required.

In final set of 93 semantic constraints, 69 in some way resembled constraints that ASPIRE generated. Of these 69, 6 were altered in a simple fashion to provide more specific feedback, 15 were either altered trivially or were based on existing constraints, and 48 made use of domain functions (see Section 2.1.2). In general, constraints which make use of domain functions could not be said to resemble constraints which ASPIRE generated. However, I used domain functions simply to apply the constraints as a whole in a different way, as opposed to fundamentally change the nature of any individual constraint. So these constraints were heavily based on those generated by ASPIRE; domain functions were required simply to change the way they were applied. This was necessary because of the way I chose to define the ideal solution. The remaining 24 constraints that make up the total 93 are considered entirely new. These 24 constraints are fairly simple, but they do not resemble any constraint which was originally generated by ASPIRE.

Development of the experimental version of Thermo-Tutor began from the completed control version. All of the alterations required to the constraints were very simple. So there was no attempt to generate the constraints for the experimental version. Instead I will describe the differences between the final sets of constraints in the control and experimental versions.

The experimental version of Thermo-Tutor contains the same set of 140 syntax constraints as the control version. The set of 87 semantic constraints in the experimental version are very similar to the 93 in the control version. From the 93 in the control version 6 were removed, 24 were trivially altered, and 12 had feedback altered to make up the 87 semantic constraints in the experimental version. The constraints which were removed were related to the "rearrangement" step which the experimental version excludes. Trivial alterations were required to 24 constraints because the representation of the ideal solution was slightly different for the third step of Thermo-Tutor. Finally, some feedback had to be altered to provide meaningful feedback for the interaction mode provided by the experimental version; formulae are "entered" as opposed to "selected".

5.2.2 Causes of these Issues

It is clear from the previous section that the domain model which ASPIRE generated was not entirely suitable for this domain. I believe there are two main reasons why this was the case. Firstly, the way I chose to define the ideal solution required many alterations to the constraints. Secondly, I believe the ontology which I developed for the third step of Thermo-Tutor was not entirely suitable for this domain.

Many of the changes to the syntax and semantic constraints were required because of the way I chose to define the ideal solution. This approach was described in more detail in Section 3.2. The problem with this approach is that it rendered false one of the fundamental assumptions that ASPIRE makes. ASPIRE assumes that the student's solution should match the ideal solution. ASPIRE generates constraints to compare the ideal solution to the student solution and vice versa. However, because of the way the ideal solution is defined, comparing the ideal solution to the student solution is not possible. This is the primary reason why many constraints were discarded. This assumption is also evident in the way ASPIRE applies constraints, or rather in the behaviour of the pattern matching functionality provided by ASPIRE. The way this pattern matching works was the primary reason why so many constraints required domain functions to operate correctly.

I believe that the ontology which I developed for the third step of Thermo-Tutor was not suitable for this domain. I believe the ontology was too simple; it was not defined with the appropriate level of granularity. I believe a more detailed, fine-grained ontology would allow ASPIRE to generate more specific constraints which could provide more detailed feedback. I defined the ontology at the level of formulae and expressions, as opposed to the level of variables and values. This meant that ASPIRE was only able to generate constraints which check formulae and expressions as a whole. More specific feedback could be provided if these constraints instead checked the smaller units of variables and values.

5.2.3 ASPIRE's Prospective Audience

The previous setions have demonstrated several difficulties that would be common when developing a constraint-based ITS with ASPIRE. It seems unlikely that ASPIRE would allow a domain expert with no programming expertise to develop an ITS for a domain as complex as thermodynamics. However, I believe there are at least two types of people that ASPIRE would be suitable for.

I believe ASPIRE would be suitable for domain experts with little or no programming expertise provided they are developing an ITS for a suitable domain. This does not necessarily mean the domain has to be very simple. The domain could contain many concepts, it is more important for the domain and task to be suitable for ASPIRE. A task which is suitable for ASPIRE could have several correct solutions. Provided that these solutions can be represented, and are distinct in that the student's solution should match one of them as a whole, ASPIRE will be able to generate constraints for this task. Provided the task is suitable, the primary limiting factor would be the domain expert's ability and understanding of domain modelling. While ASPIRE does not require programming expertise, it does require an understanding of domain modelling in order to develop the ontology. Provided a suitable ontology can be devised, the remaining steps in ASPIRE's authoring process are relatively trivial.

I believe ASPIRE is still useful in more complex domains which may not be entirely suitable. Developing an ITS with ASPIRE for such a domain will require the assistance of someone with programming expertise. However, ASPIRE simplifies the task for the programmer, and allows the domain expert to take a more active role in development.

ASPIRE provides all the same advantages to the programmer as a shell such as WETAS. It also provides the programmer with a basic ITS which can be improved. While developing Thermo-Tutor I chose to bypass some functionality provided by ASPIRE. Nonetheless, ASPIRE provided the majority of what I required. In the areas which ASPIRE fell somewhat short, it did provide a foundation from which I could build upon. Moreover, I was able to bypass certain functionality fairly easily, which shows the versatility of ASPIRE. ASPIRE allows a skilled developer to make use of its many strengths, while not being held back if certain features are not appropriate. I have no doubt that ASPIRE enabled me to achieve more during this project.

ASPIRE also allows the domain expert to take a more active role in development. In complex domains which require the assistance of a programmer it is likely that the domain expert will still be able to complete a large amount of the work themselves. For complex domains, the programmer will be required to develop the interface and edit the constraints, and may also be required to assist with developing the ontology. However, the remainder of the steps in ASPIRE's authoring process could still be completed by the domain expert.

6 Conclusions

Learning has always been a fundamental part of the human experience, and it continues to be important in modern society. The importance of learning and the need for more effective learning methods is likely to increase as we move into an age of information and technology. Many disagreements among experts in the fields of education and psychology remain, however research has produced some generally accepted results. It seems clear that the current situation in education falls desperately short of ideal, and correcting these issues will require fundamental changes in the way education is conducted in our society.

The educational needs of our society seem likely to increase because of increased reliance on information and technology. Many believe that meeting these needs will require the application of the technology itself. Computer-Based Education (CBE) presents an alternative which could help solve some of the problems in education today and in the future. Intelligent Tutoring Systems (ITSs) are among the most sophisticated applications of CBE in common use today. ITSs have the potential to provide a customized, adaptive educational experience that is pedagogically equivalent to personal one-to-one tutoring; known to be the most effective form of instruction. Significant research has been devoted to increasing the effectiveness of ITSs in various ways.

This report presents an investigation into how the interaction style of the interface influences learning in an ITS. For this purpose we used an ITS for introductory thermodynamics called Thermo-Tutor. This required the development of the standard version of Thermo-Tutor to be completed, as well as an experimental version. The standard version required input via selection, whereas the experimental version required input via typing.

The design of the experimental version of Thermo-Tutor was based on theory from psychology and cognitive science. This theory was discussed and then applied to justify the design of the experimental interface. The theories which the discussion focussed on were the generation effect, cognitive load theory and the assistance dilemma.

There were three aims of this research. The primary aim was to evaluate the effectiveness of two interfaces which employed different modes of interaction. I also aimed to evaluate the effectiveness of ASPIRE as an authoring tool, and contribute to the growing collection of ITSs.

The relative effectiveness of the two interaction modes in terms of learning remains unclear. Technical difficulties and an inappropriate group of participants severely limited the inferences that could be made from the results of the evaluation. Comparison between pre-test and post-test results could not be made. However, the results of student performance in the tutor and questionnaires were examined, and learning curves were plotted. Obtaining more conclusive results would require another evaluation to be performed with a more suitable population and implementation.

The development of Thermo-Tutor has demonstrated some of the strengths and weaknesses of ASPIRE. Some limitations of ASPIRE were evident during development. However, several strengths were also obvious, and this research also helped to identify who is likely to benefit from ASPIRE.

Development of Thermo-Tutor was generally successful because participants in the evaluation were learning. However, this research also helped to show several ways in which Thermo-Tutor could be improved. After the most crucial of these changes are completed, an evaluation could be conducted to obtain the results which could not be gathered during this project. Despite several difficulties in obtaining the results related to the primary aim of this research, the development of Thermo-Tutor using ASPIRE was successful.

Bibliography

[1] A. J. Vander, J. H. Sherman, and D. S. Luciano, *Human Physiology: The Mechanics of Body Function*. New York: Tata McGraw-Hill Publishing Company Limited, 1975.

[2] L. R. Squire, "Declarative and nondeclarative memory: Multiple brain systems supporting learning and memory," *J. Cognitive Neuroscience*, vol. 4, no. 3, pp. 232–243, 1992.

[3] X. Lu, B. Eugenio, T. C. Kershaw, S. Ohlsson, and A. Corrigan-Halpern, "Expert vs. non-expert tutoring: Dialogue moves, interaction patterns and multi-utterance turns," *LNCS*, vol. 4394, pp. 456–467, 2009.

[4] B. S. Bloom, "The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring," *Educational Researcher*, vol. 13, no. 6, pp. 4–16, 1984.

[5] M. Nichols, *E-Learning in context*. E-Primer Series, 2008.

[6] R. Zemsky and W. F. Massy, "Thwarted innovation - what happened to e-learning and why," tech. rep., The Learning Alliance for Higher Education, 2004.

[7] D. Carnevale, "Report says educational technology has failed to deliver on its promises," *Chronicle of Higher Education*, July 2, 2004.

[8] A. T. Corbett, "Cognitive computer tutors: Solving the two-sigma problem," in *UM '01: Proceedings of the 8th International Conference on User Modeling 2001*, pp. 137–147, 2001.

[9] K. VanLehn, C. Lynch, K. Schulze, J. Shapiro, and R. Shelby, "The Andes physics tutoring system: Five years of evaluations," in *Proceedings of the 12th International Conference on Artificial Intelligence in Education*, 2005.

[10] A. Mitrovic and S. Ohlsson, "Evaluation of a constraint-based tutor for a database language," *International Journal on Artificial Intelligence in Education*, vol. 10, no. 3–4, pp. 238–256, 1999.

[11] T. M. Lindquist and L. M. Olsen, "How much help, is too much help? an experimental investigation of the use of check figures and completed solutions in teaching intermediate accounting," *Journal of Accounting Education*, vol. 25, no. 3, pp. 103–117, 2007.

[12] K. R. Koedinger and V. Aleven, "Exploring the assistance dilemma in experiments with cognitive tutors," *Educational Psychology Review*, vol. 19, no. 3, pp. 239–264, 2007.

[13] A. Mitrovic, K. R. Koedinger, and B. Martin, "A comparative analysis of cognitive tutoring and constraint-based modeling," in *User Modeling 2003*, pp. 313–322, 2003.

[14] J. R. Anderson, *Rules of the Mind*. Lawrence Erlbaum Associates, 1993.

[15] S. Ohlsson, "Constraint-based student modeling," in *Student Modeling: The Key to Individualized Knowledge-based Instruction*, pp. 167–189, 1994.

[16] B. Martin and A. Mitrovic, "Authoring web-based tutoring systems with WETAS," in *Proc ICCE*, pp. 183–187, 2002.

[17] A. Mitrovic, P. Suraweera, B. Martin, K. Zakharov, N. Milik, and J. Holland, "Authoring constraint-based tutors in ASPIRE," *LNCS*, vol. 4053, pp. 41–50, 2006.

[18] A. Mitrovic, P. Suraweera, B. Martin, and A. Weerasinghe, "DB-suite: experiences with three intelligent, web-based database tutors," *Journal of Interactive Learning Research*, vol. 15, no. 4, pp. 409–432, 2004.

[19] P. Suraweera and A. Mitrovic, "An intelligent tutoring system for entity relationship modelling," *Int. J. Artif. Intell. Ed.*, vol. 14, no. 3–4, pp. 375–417, 2004.

[20] A. Mitrovic, "The effect of explaining on learning: a case study with a data normalization tutor," in *Proceeding of the 2005 conference on Artificial Intelligence in Education*, pp. 499–506, 2005.

[21] M. Uschool, M. King, S. Moralee, and Y. Zorgios, "The enterprise ontology," *Knowledge Eng. Ev.*, vol. 13, no. 1, pp. 31–89, 1998.

[22] J. Bourdeau, R. Mizoguchi, V. Psych, and R. Nkambou, "Selecting theories in an ontology-based its authoring environment," *LNCS*, vol. 3220, pp. 150–161, 2004.

[23] P. Suraweera, A. Mitrovic, and B. Martin, "The role of domain ontology in knowledeg acquisition for ITSs," in *Intelligent Tutoring Systems*, pp. 207–216, 2004.

[24] A. Bebbington, "Authoring an intelligent tutoring system for introductory thermodynamics using ASPIRE," *COSC366 Project Report*, 2009.

[25] L. L. Jacoby, "Remembering the data: Analyzing interactive processes in reading," *Journal of Verbal Learning and Verbal Behavior*, vol. 22, no. 5, pp. 485–508, 1983.

[26] N. Slamecka and P. Graf, "The generation effect: Delineation of a phenomenon," *Journal of Experimental Psychology: Human Learning and Memory*, vol. 4, no. 6, pp. 592–604, 1978.

[27] S. Bertsch, B. J. Petsa, R. Wiscott, and M. A. McDaniel, "The generation effect: A meta-analytic review," *Memory and Cognition*, vol. 35, no. 2, pp. 201–210, 2007.

[28] Z. F. Peynircioglu and E. Mungan, "Familiarity, relative distinctiveness, and the generation effect," *Memory and Cognition*, vol. 21, no. 3, pp. 367–374, 1993.

[29] B. J. Pesta, R. E. Sanders, and R. J. Nemec, "Older adults' strategic superiority with mental multiplication: A generation effect assessment," *Experimental Aging Research*, vol. 22, no. 2, pp. 155–169, 1996.

[30] D. S. McNamara and A. F. Healy, "A procedural explanation of the generation effect for simple and difficult multiplication problems and answers," *Journal of Memory and Language*, vol. 43, no. 4, pp. 652–679, 2000.

[31] J. M. Gardiner, A. J. Dawson, and E. A. Sutton, "Specificity and generality of enhanced priming effects for self-generated study items," *The American Journal of Psychology*, vol. 102, no. 3, pp. 295–305, 1989.

[32] J. Sweller, "Cognitive load theory, learning difficulty, and instructional design," *Learning and Instruction*, vol. 4, pp. 295–312, 1994.

[33] J. Sweller, J. J. G. van Merrienboer, and F. Paas, "Cognitive architecture and instructional design," *Educational Psychology Review*, vol. 10, no. 3, pp. 251–296, 1998.

[34] F. Bartlett, "Remembering: A study in experimental and social psychology.," *New York and London: Cambridge Universtiy Press*, vol. 3, pp. 1–21, 1993.

[35] M. T. H. Chi, R. Glaser, and E. Rees, "Expertise in problem solving," *Advances in the psychology of human intelligence*, pp. 7–75, 1982.

[36] W. Schneider and R. M. Shiffrin, "Controlled and automatic human information processing: 1. detection, search, and attention," *Psychological Review*, vol. 84, no. 1, pp. 1–66, 1977.

[37] R. M. Shiffrin and W. Schneider, "Controlled and automatic human information processing: 11. perceptual learning, automatic attending, and a general theory," *Psychological Review*, vol. 84, no. 2, pp. 127–190, 1977.

[38] J. S. Bruner, "The art of discovery," *Harvard Educational Review*, vol. 31, pp. 21–32, 1961.

[39] S. Papert, "Mindstorms: Children, computers and powerful ideas.," *New York: Basic Books*, 1980.

[40] L. Steffe and J. Gale, "Constructivism in education," *Hillsdale, NJ: Lawrence Erlbaum Associates, Inc*, 1995.

[41] L. J. Cronbach and R. E. Snow, *Aptitudes and instructional methods: A handbook for research on interactions*. New York: Irvington, 1977.

[42] J. Sweller, "Instructional design consequences of an analogy between evolution by natural selection and human cognitive architecture," *Instructional Science*, vol. 32, pp. 9–31, 2004.

[43] P. A. Kirschner, J. Sweller, and R. E. Clark, "Why minimal guidance during instruction does not work: An analysis of the failure of constructivist, discovery, problem-based, experiential, and inquiry-based teaching," *Educational Psychologist*, vol. 41, no. 2, pp. 75–86, 2006.

[44] K. R. Koedinger, P. Pavlik, B. M. McLaren, and V. Aleven, "Is it better to give than to receive? the assistance dilemma as a fundamental unsolved problem in the cognitive science of learning and instruction," *Cognitive Science of Learning and Instruction*, pp. 2155–2160, 2008.

[45] R. A. Schmidt and R. A. Bjork, "New conceptualizations of practice: Common principles in three paradigms suggest new concepts for training," *Psychological Science*, vol. 3, no. 4, pp. 207–217, 1992.

[46] J. Sweller, "Cognitive load during problem solving: Effects on learning," *Cognitive Science*, vol. 12, pp. 257–285, 1988.

[47] L. Vygotsky, *Mind in Society: Development of Higher Psychological Processes*. Cambridge: Harvard University Press, 1978.

[48] M. B. and M. A., "Using learning curves to mine student models," in *10th International Conference on User Modeling*, pp. 79–88, 2005.

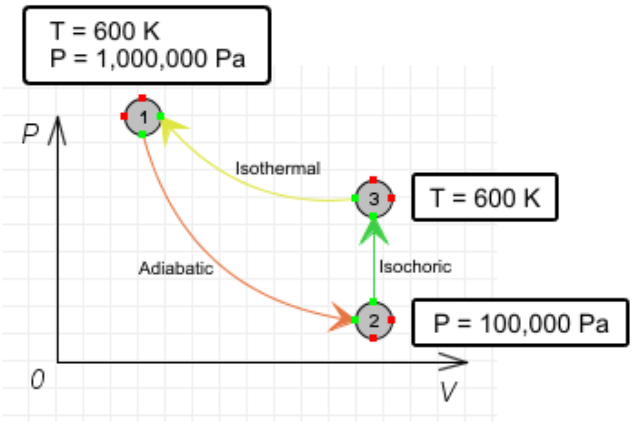
A Tests and questionnaire

Pre-Test

Your user code:.....

*Please Answer **all** questions*

1. Consider the following cycle that uses an ideal, diatomic gas as the working fluid:



Find the temperature at state 2 using one of the following formulas:

$$\frac{T_2}{T_1} = \left(\frac{P_2}{P_1}\right)^{\frac{\gamma-1}{\gamma}} \qquad \frac{T_2}{T_1} = \frac{P_2}{P_1} \qquad \frac{T_2}{T_1} = 1 \qquad \frac{T_2}{T_1} = \frac{V_2}{V_1} \qquad \frac{P_2}{P_1} = \left(\frac{V_1}{V_2}\right)^{\gamma}$$

.....

.....

.....

.....

.....

.....

.....

2. Which of the following best matches the formula for the Isothermal Work of compression/expansion of an ideal gas when n (number of moles) > 1?
The variables are not important; your task is to select the correct form

(a) $W = a \cdot b \cdot c \cdot \ln\left(\frac{d}{e}\right)$

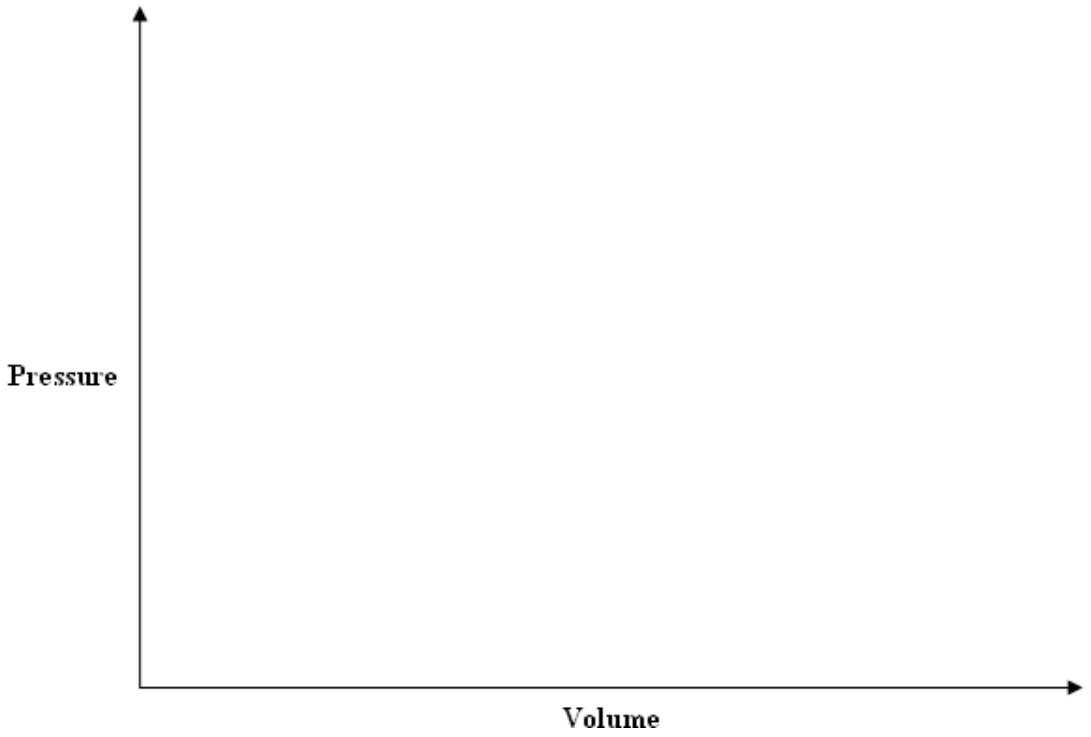
(b) $W = a \cdot b \cdot \ln\left(\frac{c}{d}\right)$

(c) $W = a \cdot b \cdot c \cdot \log_2\left(\frac{d}{e}\right)$

(d) $W = a \cdot b \cdot c \cdot \log_{10}\left(\frac{d}{e}\right)$

3 Draw a diagram of the following cycle, and specify the known properties on the diagram:

50 moles of nitrogen expands from state 1 (V = 0.3 m3) to state 2 (V = 0.5 m3) via an isobaric process (P = 400 kPa). The gas is then restored to its original volume via an isothermal compression from state 2 to state 3. Finally the gas is returned to state 1 via an isochoric pressure reduction.

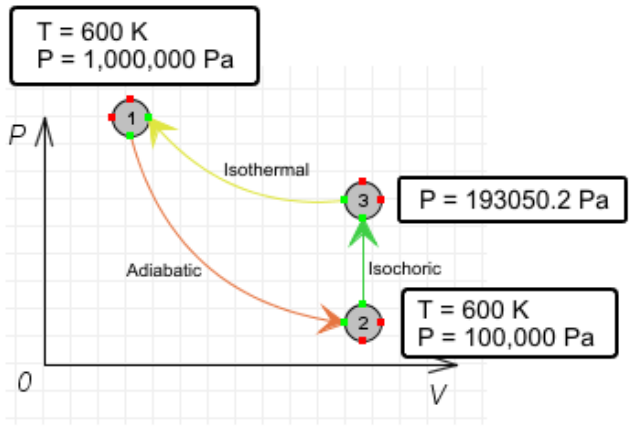


Post-Test version 1

Your user code:.....

*Please Answer **all** questions*

1. Consider the following cycle where the working fluid is 2 moles of an ideal, diatomic gas:



Find the work performed on the system during the isothermal transition using one of the following formulae:

$$W = \frac{RT_1}{\gamma - 1}\left[\left(\frac{P_2}{P_1}\right)^{\frac{\gamma-1}{\gamma}} - 1\right] \quad W = nRT \cdot \ln\left(\frac{P_2}{P_1}\right) \quad W = RT \cdot \ln\left(\frac{V_1}{V_2}\right) \quad W = \frac{nRT_1}{\gamma - 1}\left[\left(\frac{P_2}{P_1}\right)^{\frac{\gamma-1}{\gamma}} - 1\right]$$

$$W = nRT \cdot \ln\left(\frac{V_1}{V_2}\right) \quad W = RT \cdot \ln\left(\frac{P_2}{P_1}\right)$$

2. Which of the following best matches the formula for the relationship between the ratio of final to initial temperature and the ratio between final and initial pressure for an adiabatic, compression/expansion of an ideal gas?

   The variables are not important; your task is to select the correct form

   (a) $\frac{a}{b} = \frac{c}{d}$

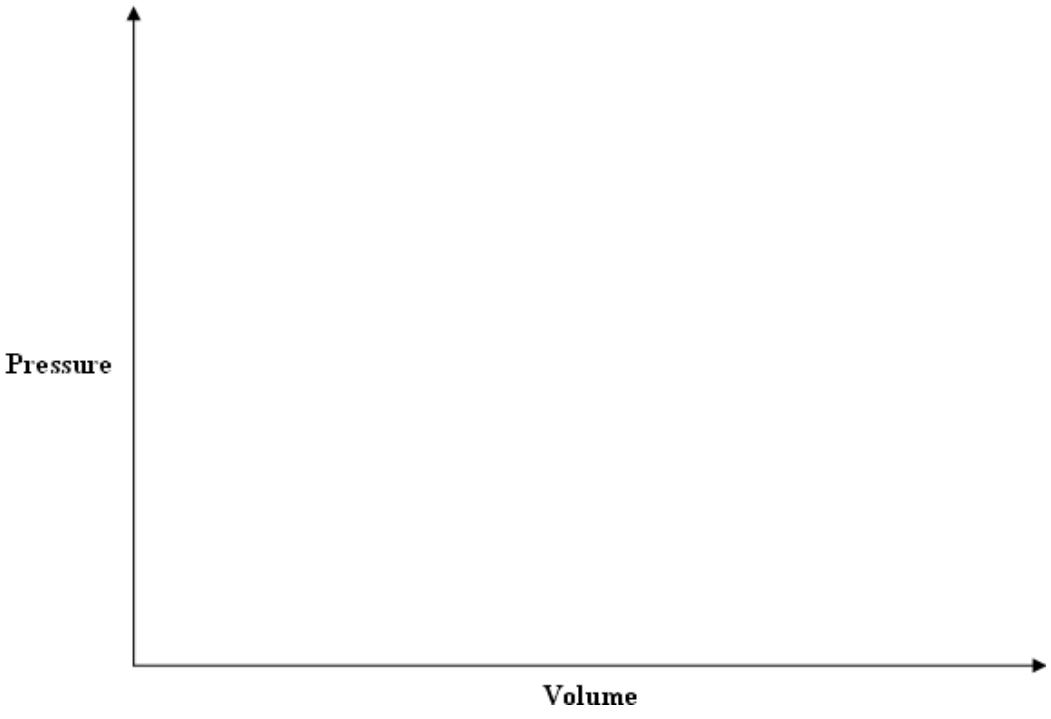
   (b) $\frac{a}{b} = \left(\frac{c}{d}\right)^{e}$

   (c) $\frac{a}{b} = \left(\frac{c}{d}\right)^{\frac{e-1}{f}}$

   (d) $\frac{a}{b} = c$

3 Draw a diagram of the following cycle, and specify the known properties on the diagram:

One mole of an ideal, diatomic gas is compressed adiabatically from 5 bar, 300 K (state 1) to 10 bar (state 2). The gas is then reduced in pressure at constant volume back to 5 bar (state 3). Finally the gas is isobarically expanded back to state 1.



Post-Test version 2

Your user code:..... Please Answer **all** questions

1. Which of the following best matches the formula for the relationship between the ratio of final to initial temperature and the ratio between final and initial pressure for an adiabatic compression/expansion of an ideal gas?
   The variables are not important; your task is to select the correct form

   $\frac{a}{b}=\frac{c}{d}$ $\frac{a}{b}=\left(\frac{c}{d}\right)^{e}$ $\frac{a}{b}=\left(\frac{c}{d}\right)^{\frac{e-1}{f}}$ $\frac{a}{b}=c$

2. Which of the following best matches the formula for the Isothermal Work of compression/expansion of one mole of an ideal gas?
   The variables are not important; your task is to select the correct form

   $W=a\cdot b\cdot c\cdot\ln\left(\frac{d}{e}\right)$ $W=a\cdot b\cdot\ln\left(\frac{c}{d}\right)$ $W=a\cdot b\cdot c\cdot\log_2\left(\frac{d}{e}\right)$ $W=a\cdot b\cdot c\cdot\log_{10}\left(\frac{d}{e}\right)$

3. Write the formula for the relationship between the ratio of final to initial temperature and the ratio between final and initial pressure for an isochoric process involving an ideal gas.

   .....

   .....

4. Write the expression relating the change in internal energy (ΔU) to the change in temperature (ΔT) between two states for n (number of moles) > 1

   .....

   .....

5. Which of the following best matches the formula for the Adiabatic Work of compression/expansion when n (number of moles) > 1?

   $W=\frac{nRT_1}{\gamma-1}\left[\left(\frac{P_2}{P_1}\right)^{\frac{\gamma-1}{\gamma}}-1\right]$ $W=\frac{RT_1}{\gamma-1}\left[\left(\frac{P_2}{P_1}\right)^{\frac{\gamma-1}{\gamma}}-1\right]$ $W=nRT\cdot\ln\left(\frac{P_2}{P_1}\right)$ $W=RT\cdot\ln\left(\frac{P_2}{P_1}\right)$

6. Which of the following best matches the formula for the Isobaric Work of compression/expansion?

   $W=-P(V_1-V_2)$ $W=-R(V_2-V_1)$ $W=-R(V_1-V_2)$ $W=-P(V_2-V_1)$

Questionnaire

1. How would you rate the overall quality of Thermo-Tutor?

1---------- 2 ---------- 3 ---------- 4 ---------- 5

Poor Excellent

2. Rate your impression of Thermo-Tutor:

1---------- 2 ---------- 3 ---------- 4 ---------- 5

Low Mental Effort Required High

1---------- 2 ---------- 3 ---------- 4 ---------- 5

Low Problem Difficulty High

1---------- 2 ---------- 3 ---------- 4 ---------- 5

Low Enjoyment High

3. How would you rate the quality of the feedback from Thermo-Tutor?

1---------- 2 ---------- 3 ---------- 4 ---------- 5

Poor Excellent

4. What did you like about Thermo-Tutor?

.....

.....

.....

.....

5. What changes would you like to see in Thermo-Tutor?

.....

.....

.....

.....

6. Any other comments?

.....

.....

.....

.....