PhD Dissertation Proposal

Impact of Proactive Scaffolding and Elaborated Cognitive Model on Learning Geometry Theorem Proving with Construction

Noboru Matsuda

Intelligent Systems Program

University of Pittsburgh

January 26, 2004

Abstract: I propose to build an intelligent tutoring system that provides scaffolding not only on the problem-solving steps but also on the inference steps underlying each problem-solving step. Such inference steps are usually processed mentally by experienced problem solvers and merely taught explicitly to students. A key inference step is to transform textbook knowledge (geometry postulates), which is usually provided in a descriptive form, into a conditional form, which distinguishably shows the premises and a consequence of the postulates. The tutor starts a tutoring session from *telling* students how to solve problems by showing all those inference steps prior to *asking* them to solve problems by themselves. The tutor then gradually declines the degree of scaffolding and let students do more. Besides the novelty of the tutoring strategy, an originality of the proposed study is in its domain, namely, geometry theorem proving with construction. Theorem proving with construction is one of the most challenging subjects for students to learn. The best we have known so far is to teach them heuristics to encourage their "creative" thought. The major intellectual interest of the proposed study is to see an effectiveness of the designed tutoring strategy for students learning geometry theorem proving with construction.

Contents:

1 Introduction ... 3

2 Issues on Learning and Teaching Geometry Theorem Proving ... 4

2.1. Student's Difficulty in Learning Proof Writing ... 4

2.2. Theoretical Implications in Teaching Geometry Theorem Proving ... 5

2.2.1. Articulating tacit knowledge ... 6

2.2.2. Teaching operationalization ... 7

2.2.3. Proactive scaffolding ... 7

2.3. Summary of a Learning Environment for Geometry Theorem Proving ... 8

3 Proposed Dissertation Study ... 8

3.1. Research Questions ... 9

3.2. Methodology of Investigation ... 10

4 The Advanced Geometry Tutor ... 11

4.1. Brief Review of AGT ... 11

4.2. Cognitive Task Analysis for Theorem Proving ... 12

4.3. Scaffolding Strategy ... 15

4.4. Learning Environment ... 17

5 Evaluation ... 19

5.1. Procedure ... 20

5.2. Material ... 21

5.3. Hypothesis ... 22

6 Milestone ... 25

7 Intellectual Benefits ... 26

1 Introduction

I propose a teaching strategy for an intelligent tutoring system (ITS for short) in geometry theorem proving with construction. Besides that this is apparently the first attempt to build an ITS for geometry theorem proving that requires construction, the novelty is at the way the tutor provides scaffolding for novice students. The proposed ITS has an elaborated cognitive model of theorem proving and explicates expert's inference steps that are usually unobservable mental activities hence are not subject of direct tutoring. Having such a fine grained cognitive model, the major impact on instruction is believed to arise from two aspects. First, the tutor starts a tutoring session with *showing* students how to solve problems before *asking* them to solve problems by themselves. Such scaffolding is called *proactive scaffolding*. Second, the elaborated cognitive model even involves inference steps to identify the conditions and a consequence of a postulate application[1], which corresponds to a single proof step. Such inference steps are called *operationalization* of a postulate that is usually taught in a declarative form.

This simple form of tutoring strategy involves several interesting aspects of cognitive theories on teaching and learning. The ultimate goal of the current dissertation study is to investigate the effectiveness of such tutoring strategy by analyzing students' learning outcomes. More specifically, we build an intelligent tutoring system for geometry theorem proving with construction that realizes the proposed tutoring strategy, and test its impact on students' learning. The proposed ITS is called *Advanced Geometry Tutor* (AGT for short).

---

[1] In this document, the word "postulate" is used for those things that can be used as a justification of proposition in a proof. Namely, definitions, axioms, and theorems are all called postulates.

This proposal first discusses an overview on teaching and learning geometry theorem proving. The overview first shows students' difficulties in learning geometry theorem proving. It then discusses cognitive theories on teaching and learning theorem proving. Some existing tutoring systems on theorem proving are also described. Section 3 then claims research questions and methodology of the proposing study. Section 4 shows the architecture of AGT. The description of the tutor includes a cognitive task analysis for geometry theorem proving, proactive scaffolding strategy, and the individual components in the learning environment. Section 5 is about the evaluation to address the research questions mentioned in Section 3. A chronological estimation to accomplish current dissertation study is provided in Section 6. Finally in Section 7, the proposal conjectures contributions of the current study to the research communities including intelligent tutoring systems and learning science.

2 Issues on Learning and Teaching Geometry Theorem Proving

This section provides a review on students' difficulties in learning proof writing as well as theoretical implications for designing a tutoring system.

2.1. Student's Difficulty in Learning Proof Writing

Geometry theorem proving appeared to be one of the most challenging subjects for students. In a large scale classroom evaluation with 1520 students, Senk (1985) showed that barely 20% of the students could do proofs with great complexity at the end of a year-long geometry class. 30% showed mastery in proof writing but for the problems that are similar to the ones in textbook, 25% could only do trivial proofs.

Especially remarkable issue is that the students fail to write proofs even when they are competent for concepts on geometric propositions and postulates. Koedinger (1990) reported that the students had only 35.5% accuracy on proof writing at a pre-test even though they showed 67% accuracy on the test items for judgment of geometric statements. This implies that applying geometric concept to *justify* geometric statements (i.e., to write a proof) is more difficult than to *judge* them (i.e., to say a statement is correct or it is uncertain). Chaiyasang (1989) showed that less than 15% of the students could achieve "good" in proof writing even though they are ranked as the van Hiele level 4, which means that they understood geometric concepts necessary to write a proof.

These studies show that it is not enough to teach geometric concepts to have students a masterly of proof writing. No wonder there are very many studies on teaching proof writing that failed to show a significant effect. Students apparently need deliberate practice in writing proofs, but the question is how such practice looks like.

2.2. Theoretical Implications in Teaching Geometry Theorem Proving

Geometry Proof Tutor is one of the most successful tutoring systems on geometry theorem proving (Anderson, Corbett, Koedinger, & Pelletier, 1995). It utilized so called *model tracing* technique that captures a model of theorem proving as a set of production rules. It is a piece of production rules that is a unit of instruction for the model tracing tutors. The tutor monitors whether the student can apply a particular production rule in a particular situation, and if he shows a failure, then the tutor gives an instruction on how to apply the targeted production rule.

Even a model tracing tutor could not bring students as much learning gain as human tutors do. Since a model tracing tutor is built upon the production system framework, the tutor can only teach cognitive skills that are represented by an underling cognitive model. ACT-R theory of learning, on which the model tracing tutors are designed, advocates that having repetitious practice of a cognitive skill would eventually lead a learner to mastery of the skill. It would then be interesting to see whether one can achieve mastery on any cognitive skills given that such skills can be organized as a coherent cognitive model. Geometry Proof Tutor models the backward inference as three cognitive steps: (1) to identify a goal to prove, (2) to identify a production rule to apply, and (3) to identify subgoals that are conditions of the production rule. What if we build more elaborated cognitive model of proof writing? Does it help students to learn more? And does it even reduce the learning time? These are the questions that motivated the current study.

Based on some cognitive theories on learning problem solving, we suppose that the following instruments would have a significant impact on students' learning: (a) articulating tacit inference steps, (b) operationalization, and (c) proactive scaffolding. The following sections discuss those issues.

2.2.1. Articulating tacit knowledge

Effect of learning with worked-out examples can be seen in many studies (see for example, Sweller & Cooper, 1985). However, as mentioned in VanLehn et al. (1992) a great amount of steps taken to solve a problem could be tacit in worked-out examples. This is why asking students to self-explain worked-out examples would further increase the learning gain (Chi, Bassok, Lewis, Reimann, & Glaser, 1989). A cognitive task analysis of geometry theorem proving shown in Section 4 indeed demonstrates that only 44% (7 out of 16) of the inference steps corresponds to observable operations onto some GUI component (i.e., to press a button, to enter an equation, etc). Those hidden steps might be a source of the failure of learning for poor students who can not reveal such tacit knowledge even when they do self-explain problem-solving steps.

Hence we first claim that explicating unobservable mental manipulation taken by expert problem solvers encourages learning theorem proving. One can build a cognitive model of problem solving at an extremely find grain size including all perceptual and motor skills. We have yet to know what exactly the right grain size for such a cognitive model is. The next section discusses one of the cognitive skills that have been reported critical for proof writing.

2.2.2. Teaching operationalization

One of a hidden inference steps that have been occasionally reported to be particularly important for proof writing is to transform geometry statements written in a declarative form into a conditional form. For example, a theorem taught in a declarative form "two base-angles of an isosceles triangle are equal" must be translated into a conditional form "if a goal is to justify ∠ABC=∠ACB in a triangle ABC, then hypothesize that AB = AC and justify it." We hereafter call this transformation *operationalization*.

A number of studies have shown a positive impact of teaching operationalization. Tubridy (1992) found that teaching geometric postulate in an if-then statement helps students at the low and the middle levels. Greeno (1983) examined an impact of teaching so called proof-checking procedure that emphasizes to decompose an inference into conditional and consequential parts. His experiment showed that when students were taught the proof-checking procedure, their performance in proof writing significantly increased.

These studies suggest that students need to learn operationalization explicitly as a part of the cognitive skills for proof writing. Hence we claim that the underlying cognitive model of proof writing should include operationalization as a part of the inference steps.

2.2.3. Proactive scaffolding

Most tutoring systems ask students to solve problems. These tutoring systems assume that the students have learned knowledge necessary to solve problems and that what they need is to stabilize their knowledge through practice. However, as VanLehn (1998) mentioned, when the students reach an impasse, they often need to go back to the examples to do an analogical reasoning by reading steps off the examples. Most of the computerized tutors do not allow students having this kind of retrograde reference to the examples.

Based on the underlying cognitive model of proof writing, the tutor can perform all the inference steps no matter if they are observable or not, and show students what they are supposed to do for each step. We call this type of scaffolding *proactive scaffolding*. At the beginning of the tutoring session, providing proactive scaffolding would equally effective as letting students read a worked-out examples. When a student gets stuck, invoking proactive scaffolding has the same effect to referring back to examples.

We then claim that proactive scaffolding is necessary for students who are not yet reaching a mastery of proof writing. We further suppose that this kind of scaffolding must be fed away as learning proceeds, but it also must be fed *in* when students show poor performance.

2.3. Summary of a Learning Environment for Geometry Theorem Proving

The target students in this study are at an intermediate level who know geometry knowledge necessary to write some proofs and learn more postulates, but their knowledge is not fully stabilized hence they may need to refer worked-out examples during problem solving. The survey in the previous section suggests that an ideal learning environment for these students should provide proactive scaffolding that fades away as the student get familiar with proof writing, but also fades in when necessary. Proactive scaffolding must reify problem-solving steps in great details including the unobservable inference steps.

3 Proposed Dissertation Study

This section compiles the research topics mentioned in the previous sections into research questions and provides a methodology of the dissertation study.

3.1. Research Questions

Our primary interest is on an effectiveness of Advanced Geometry Tutor in learning proof writing. Since learning involves various aspects that interact with each other, we focus on some specific aspects of cognitive-skill learning on which our tutor may have a significant impact. Specifically, we investigate the three aspects described below.

***Impact on learning to learn*:** Anderson, Bellezza, and Boyle (1993) fond that students find easier to learn new postulate over the first chapters of a geometry course. The students' performance on applying a novel postulate is better at the later stage of learning than the beginning. They apparently had learned to learn an application of new postulate.

A cognitive model of a postulate application described in Section 4.2 involves both *content general* and *content specific* inference steps. Such difference in the constituents may account for students' learning to learn.

The content general inference steps require students to do the same operation independent of the postulate's content (e.g., to select an unjustified proposition to justify next), whereas for the content specific ones, students must learn different operations for different postulates (e.g., to rephrase a postulate in a conditional form). The students apparently encounter more occasions to practice content general inference steps than content specific ones. Hence, it might be the case that learning of new postulate occurs in different speed between content general and content specific inference steps. In other words, students' performance on the content general steps becomes faster at later time hence overall performance on a novel postulate increases. We would verify this claim by measuring individual learning curves on each inference steps.

Furthermore, since out tutor explicitly teaches all inference steps it may affect students' learning to learn new postulates. The first research question is then:

- Does providing proactive scaffolding on the elaborated inference steps have influence on learning new postulates?

***Impact on learning individual postulates*:** The effectiveness of our tutor can be measured as a speed-up in learning curve for individual postulates. Namely, if our tutor effectively expedites student's learning for a postulate, then we should observe a steep drop off in the learning curve at the beginning for all new postulates.

- Does providing proactive scaffolding on elaborated inference steps brings students quick improvement on learning postulates?

***Impact on individual difference*:** So far, we have discussed learning gain as the aggregated phenomena. Namely, both the speed and the accuracy of postulate applications are measured in the aggregate of the students and the occurrences of postulate applications. However, different students would benefit differently from the tutor. Hence there would be a group of students that the tutor can help a lot and another group of students that the tutor is not effective. To see the effectiveness of the tutor across the students' individual differences, we compare the students' performance within an experiment group.

- How does the tutor work across the students' population?

3.2. Methodology of Investigation

To answer the questions mentioned above, we build two versions of tutors: a *full* tutor fully provides intensive scaffolding upon the elaborated cognitive model whereas a *restricted* tutor suppresses scaffolding for the steps that are usually tacit. We then run evaluation with two conditions assigned to one of two tutors, and compare learning gains.

The details of Advanced Geometry Tutor are provided in the next section, and detailed experimental design for the tutor evaluation is described in Section 5.

4 The Advanced Geometry Tutor

This section describes the architecture of Advanced Geometry Tutor and the details of proactive scaffolding that the tutor provides. The explanations in this section would be best understood by referring a snapshot of the AGT learning environment shown in Figure 4 (p.17).

4.1. Brief Review of AGT

AGT teaches how to compose a two-column proof. A two-column proof is represented as a table where each row shows a proposition, which is a geometric assertion, appeared in the left column while supported by a justification in the right column. A proposition is justified in one of three ways: (1) typing a word "Given" in the justification cell for a given proposition, (2) typing a word "Identity" in the justification cell when a proposition is about an equality of identical elements (e.g., $\angle ABC = \angle ABC$), or otherwise (3) entering a name of the postulate that has a consequence that matches with the proposition. For the last case, premises of the postulate must be also entered. Line numbers are assigned to each row so that the premises are represented by the line number. At the beginning, the proof table only contains given propositions at the top rows with "Given" as the justification, followed by a goal in the adjacent row with an empty justification cell. Students are supposed to fill up the empty cells in the proof table so that all propositions have valid justifications.

AGT only teaches backward inference as a proof strategy. When the students are using the tutor, they are only allowed to follow an exact sequence taught by the tutor. AGT provides proactive scaffolding when a student is in need, while minimally interferes as the student becomes more competent. At the beginning of a tutoring session, when the tutor does not know the student's competence level on proof writing, AGT shows the student how to solve a problem, namely, how to fill up a proof table. The tutor shows all inference steps in the Inference Step window along with the explanations of each inference step shown in a separate message window. At any time of the tutoring, the Inference Step window shows inference steps to make a single postulate application that corresponds to fill up a single row in the proof table (both a proposition and a justification). The steps in the Inference Step window are properly indented so that one can see the step-substep relationships involved in a backward postulate application.

As the tutoring session progresses and the tutor recognizes that the student's competence level on a particular inference step is above a threshold, the tutor shifts scaffolding strategy from proactive to modest. Under the modest scaffolding mode, the tutor only prompts the student a next step and asks the student to perform it. When the student's competence level further goes up, the tutor completely turns off scaffolding. When the student makes an error on any inference steps, the tutor immediately provides a feedback and forces the student to fulfill a correct inference step. When the student gets stuck, he/she can request a help. The tutor then shows what to do next.

The tutor maintains the student's competence level on individual inference steps as a numerical variable. When the student performs a step correctly, then the competence level of that step increases. When the student can not perform a step correctly, then the competence level of that step decreases. When the competence level happens to fall below a threshold, then the tutor resumes proactive scaffolding.

4.2. Cognitive Task Analysis for Theorem Proving

A cognitive task analysis has been taken place to build a model of backward inference for proof writing. Our previous study on GRAMY, an automated geometry theorem prover that can find constructions, revealed that for the most of the problems used in textbook, construction does not need any peculiar procedure but can be done by a slightly modified backward postulate application (Matsuda & VanLehn, 2004). AGT teaches this technique to the students.

The model of theorem proving represents a hierarchy of goals. The leaves in the hierarchy are inference steps that do not have subgoals. Each of them corresponds to an operation that is either an observable manipulation on a proof table or an unobservable mental manipulation (e.g., to see if a proposit ion to be asserted is already in the proof table). Since a backward inference that requires construction is slightly different from the one that does not require construction, AGT has two cognitive models.

Figure 1 shows a cognitive model of a backward inference that does not involve constructions. The italicized leaf steps are the observable steps. A backward inference consists of two major goals: “Select a proposition to justify” and “Apply a postulate backward.” The former is an operation to select an unjustified proposition in the proof table. The latter step has two substeps, “Select a postulate” and “Execute the postulate.” Both substeps are further broke down.

- Backward inference
  - ***Select a proposition to justify***
  - Apply a postulate backwards
    - Select a postulate
      - ***Pick a postulate***
      - Transform the postulate into a conditional form
      - Overlap configurations
    - Execute the postulate
      - Instantiate premises
      - Check Duplication
      - ***Assert premises as unjustified propositions***

Figure 1: Cognitive model of a backward inference without construction

The step “Pick a postulate” is an operation to enter a postulate name into a justification cell next to the selected unjustified proposition. The step “Transform the postulate into a conditional form” is a realization of operationalization discussed in Section 2.2.2. The step “Overlap configuration” is to overlap a postulate onto the problem figure. If, for example, the student picked the postulate on triangle congruent then two triangles must be overlapped onto the problem figure so that the postulate's consequence matches with the selected unjustified proposition. This step is necessary to figure out the premises for the succeeding steps. The last two steps mentioned here have nothing to do with actual manipulation onto the proof table, but the tutor pops up a dialogue window and ask the student to enter an appropriate response.

Finally, there are three substeps for the step "Execute the postulate." The first substep asks students to identify premises necessary to apply the selected postulate. This step corresponds to a mental manipulation to figure out premises to be asserted into the proof table, but not actually making any assertion at this point. The second substep "Check duplication" asks to exclude premises that are already in the proof table. The last substep is an operation to assert the premises into the proof table as unjustified propositions.

Figure 2 shows a cognitive model of a backward inference with construction. The basic idea is that a construction is done so that a selected postulate can be applied. The selected postulate has a consequence that matched with the proposition to be justified hence overlapping of the configuration can be partially done. The construction then must be carried out so that the missing segments are added. In the proposed cognitive model, the construction then takes place between "Select a postulate" and "Execute the postulate." The method for construction consists of two substeps: "Find a partial match" and "Construct missing segments." Remaining inference steps are identical to the ones for a backward inference without construction.

Backward inference construction
  Select a proposition to justify
  Apply a postulate backwards
    Select a postulate
      Pick a postulate
      Transform the postulate into a conditional form
      Overlap configurations
    Construction
      Find missing segments
      Construct missing segments
    Execute the postulate
      Instantiate premises
      Check Duplication
      Assert premises as unjustified propositions

Figure 2: Cognitive model of a backward inference with construction

4.3. Scaffolding Strategy

Figure 3 shows the procedure to provide scaffolding. Given a problem, the tutor first generates a *linear solution tree*, which is a sequence of postulate application involved in a proof. The tutor generates a linear solution tree by traversing a search space depth-first. Nodes in a linear solution tree holds (a) a current goal to prove, namely, a proposition to justify, (b) a goal queue, which is a list of unjustified propositions, (c) a goal stack, which is a list of unjustified propositions being justified, and (d) subgoals to achieve the current goal.

To provide scaffolding, nodes in the linear solution tree are further broken down into inference steps shown in Figure 1 or Figure 2 depending whether it requires construction. Given a node in the linear solution tree, the tutor provides scaffolding on each inference steps by invoking the procedure called `scaffolding-inference-step` in Figure 3. Depending on the student's competence level on each inference step, the tutor provides different degree of scaffolding. If the student does not demonstrate competent performance on the target inference step, then the tutor provides proactive scaffolding by articulating the step. If the student shows moderate competence, then the tutor just prompts the inference step and let the student fulfill the step. If the student shows full competence, then the tutor keeps silent.

Student model for AGT has not been fully specified at this point. AGT would have a simple student model that represents the student's competence level for each inference step for each postulate. The competence may be represented as a numerical value that is increased when the student shows a correct performance and decreased when he failed on the step.

```
Procedure scaffolding( problem )
     solution-steps ← a linear solution tree for problem
     for each step in solution-steps do
          if  requires construction do
               scaffolding-inference-step( “backward inference construction” )
          else do
               scaffolding-inference-step( “backward inference” )

Procedure scaffolding-inference-step( inference-step )
     competence-level ← the student’s competence level for inference-step
     if competence-level < 2 do
          articulate inference-step
          if inference-step has substeps do
               sub-inference-steps ← inference steps required to achieve inference-step
               for each substep in sub-inference-steps do
                    scaffolding-inference-step( substep )
          else do
               scaffolding-operation( inference-step )
     else if 2 =< competence-level < 3 do
          prompt inference-step
     else if 3 =< competence-level do
          nothing

Procedure scaffolding-operation( inference-step )
     if the student’s competence level for inference-step is less than 1 do
          articulate inference-step
     else
          prompt inference-step
```

Figure 3: Scaffolding Strategy

4.4. Learning Environment

The Advanced Geometry Tutor consists of several GUI components to help students learn how to compose a proof. Figure 4 shows a screen shot of the tutor. On the left side from top to bottom, the tutor provides Message window, Problem Description window, and Proof Table window. On the right hand side, there are Inference Steps window and Postulate Browser window. Brief descriptions for each window follow.

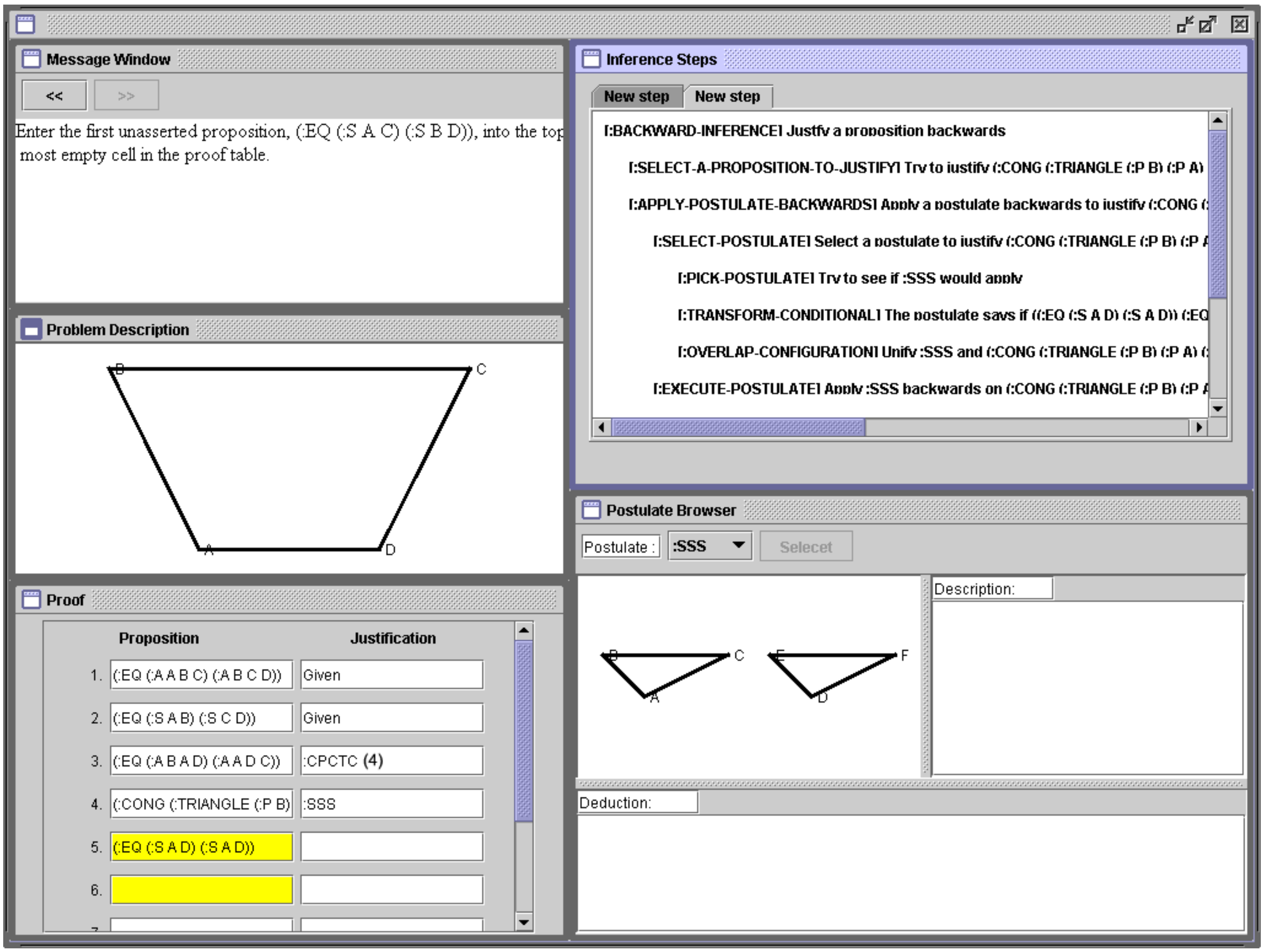

Figure 4: The Advanced Geometry Tutor learning environment (must be replaced)

Message window: Any kind of message from the tutor appears in this window. This window is also used for the inference steps that do not have direct manipulation onto any GUI component. It displays a message from the tutor and let the student enter a response. When a student makes an error, the tutor provides a feedback here. Students must acknowledge to the feedback by clicking [OK] button to proceed to the next action. Dialogues are stored and free to browse back and forth by clicking a backward [<<] and a forward [>>] button.

Problem Description window: This window shows a problem statement and a problem figure. The problem figure displayed in this window is used in two ways: (1) highlight geometric elements appeared in a proposition used in other GUI component and (2) draw additional segments onto the problem figure for construction.

Proof window: A proof is realized as a two-column table where each row consists of a proposition and its justification. A justification consists of a name of postulate and its premises that justify the proposition. The premises are represented as a list of line numbers for the corresponding propositions.

Inference Step window: The Inference Step window reifies the cognitive model of backward inference as an indented text where each line corresponds to a single inference step. At the beginning, the tutor only shows the top-most inference step, which is either “Backward inference” or “Backward inference construction” (see Figure 1 and Figure 2). A student is asked to click on that inference step, and its substeps appear under the top-level inference step. The student is then asked to click the inference step that he must perform next. The tutor displays its substeps underneath. This process is repeated until the student expands all the inference steps. At the time the student expands all the inference steps, a single postulate application is done, which means that a single row in the proof table is filled out and some new propositions are asserted with an empty justification. At this point, the tutor clears the Inference Step window and displays a new top-level inference step for the next proof step.

Postulate Browser window: The students can refer to the postulates they are supposed to use in a proof. When one clicks a name of a postulate listed in the browser window, a typical configuration for the postulate as well as its premises and a consequence would be displayed.

There is another window that is not shown in Figure 4. Since geometry equation may involve special symbols that are not appeared on the keyboard (e.g., Δ, ∠, ≡, etc), the tutor provides students with ***Proposition Builder*** that allows them to compose a geometric equation. The Proposition Builder window appears automatically when needed.

5 Evaluation

This section describes experimental design for an evaluation of AGT to explore an answer for the research questions mentioned in Section 3.1. We use following dependent variables to measure learning gains: (1) speed and accuracy of applying new postulates, (2) chronological improvement (in the speed and the accuracy) on individual postulate applications, and (3) number of students that show significant improvement in the learned postulate applications.

For all three research questions, we compare an experimental group, which uses AGT, and a control group, which uses restricted version of the tutor. The *AGT group* uses a full version of AGT that provides proactive scaffolding described in Section 4. The *restricted-AGT group* uses a modified version of the tutor that suppresses scaffolding in the following way: (1) the Inference Step window does not appear, and (2) the tutor only provides scaffolding on inference steps that is observable hence is associated with some kind of operation with the graphic user interface. More precisely, the restricted tutor only provides scaffolding on steps that are italicized in Figure 1 and Figure 2.

Besides the difference in the appearance of Inference Steps window and underneath cognitive model, remaining things are the same in both conditions. The tutors in both conditions first show students how to compete a proof (i.e., provide proactive scaffolding for an entire problems solving process). When the students are asked to solve problems by their own, no matter which tutor they are using, they must follow the exact sequence of backward inference demonstrated by the tutor. When students make an error, both tutors provide immediate feedback and force students to perform correct operation. Students can ask a help when they reach an impasse.

We use another learning group as a control condition for both of the tutored groups. Participants in the third group learn proof writing in an ordinary classroom fashion. Namely, they read an instructional material that explains how to prove geometry theorems, solve some proof problems in a workbook, and receive some feedback from a teacher with a written comment on their works.

In short, the experiment is (1) to compare the impact of reifying and not reifying inference steps as a visual clue to make a backward inference, (2) to compare the impact of having fine grained cognitive model, which captures not only observable operations but also unobservable inference steps, and having a coarse mode, which only captures observable operations, and (3) to compare those tutoring conditions with an ordinary paper-and-pencil learning condition.

5.1. Procedure

All participants take the Van Hiele Geometry Test (Usiskin, 1982). Participants are then randomly assigned to one of the learning conditions (AGT, restricted-AGT, or paper-and-pencil), while the average pre-test scores are counter balanced so that all groups have an equal number of subjects at an equal competence level.

The participants in AGT and restricted-AGT group follow same procedure. First, they watch a demonstration of the assigned tutor to learn how to use it. Next, they work on two practice problems. Then, they have a tutoring session where the students are asked to solve 4 problems. Finally, as a post-test, they solve two assessment problems. There is a new postulate introduced to solve both problems. For the second problem, an additional new postulate is introduced. The first assessment problem does not require construction hence the new postulate can be applied straightforward. On the other hand, the second problem requires a construction to apply the new postulate. During the tutoring and assessment sessions, the students' activities are logged to measure the time to complete each step and the accuracy of the step.

The participants in the paper-and-pencil group are provided an instructional material that explains how to write a proof. The same problem used in the demonstration in the tutored conditions is used in the instruction (the problem D1-125 shown in Figure 5). Next they take a couple of "practice" for proof-writing. In the practice, the students follow a written instruction to complete a proof. After the practice, the students solve 4 proof problems and hand them in to the human tutor. The human tutor grades the students' answers and returns them to the students. Finally, the students take a post-test with two problems that are the same problem used in the tutored conditions. The post test is also written on a paper and the students are supposed to write a proof with a pencil.

5.2. Material

To measure a speed-up in learning a new postulate application, we order problems so that each problem requires a few (at most 2) new postulates to learn. Figure 5 shows the problems used in the evaluation. It also shows the chronological order of the problems. The top row shows the names of the postulates and the construction procedures. The construction procedure C-CP means to connect existing two points and C-EX means to extend an existing segment. The o's and x's in the table indicate that the corresponding postulate is required to apply to solve the problem shown at the left column. A 'x' shows that the application of corresponding postulate requires construction. Appendix shows all postulates used in the experiment and their declarative as well as the operationalized version of descriptions.

	Problem	Postulate								Construction	
		CPCTC	SelfCong	SAS	VerAng	Z	ASA	Mtri	Trans	C-CP	C-EX
Demonstration	D1-125	o	o	o							
Practice	D2-125	o	o	o							
	P123	X	o	o						X	
Tutoring	D1-123	o		o	o						
	D9-148*	o			o	o	o				
	12.4.1*	o			o	o	o				
	P132	X	o	o		o				X	
Assessment	P110**	o			o	o	o	o			
	P110*	o			o	o	o	X	o	X	X

Figure 5: Problems used for the evaluation

5.3. Hypothesis

Now we discuss our hypotheses for the research questions mentions in Section 3.1.

Impact on learning to learn: We hypothesize that the AGT group shows rapid drop in the learning curve on the time and accuracy of the *first* application of newly introduced postulates. Figure 6 shows a conjecture of the appearance of two learning curves. It plots performance (could be speed or accuracy) on the first application of a newly introduced postulate. Data collected from all students are aggregated. We conjecture that the AGT group would learn new postulates faster and more accurate than the control group, and the gap between two groups would increase as learning proceeds.

As discussed earlier, this conjecture is based on the observation that the students in the AGT group explicitly learns unobservable inference steps that fill up 57% of the content general inference steps, whereas the students in the control group must inductively learn these steps by themselves. Hence there would be a great chance that AGT facilitates learning to learn.

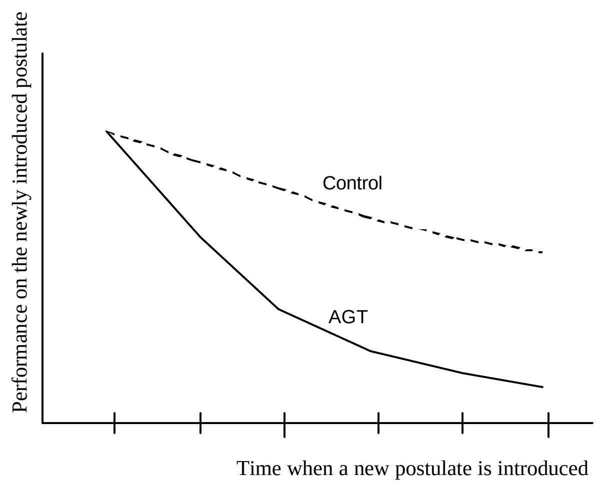

Figure 6: Learning curve on the performance of the first postulate application

Content general	Content Specific
Select a proposition to justify	Transform the postulate into a conditional form
Apply a postulate backwards	Overlap configurations
Select a postulate	Instantiate premises
Pick a postulate	
Execute the postulate	
Check Duplication	
Assert premises as unjustified propositions	

Figure 7: Cognitive skills used in backward inference

Impact on learning individual postulates: We hypothesize that even though the experiment group might reach masterly in individual postulate applications faster than the control group, the effect is not huge. After rather a few practices, both conditions might achieve at the same performance level.

Figure 8 shows the learning curve on postulate applications that is aggregated on both the students and the postulates.

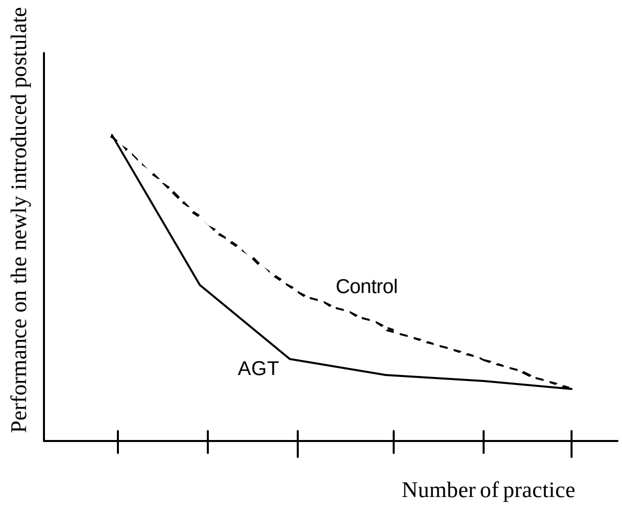

Figure 8: Learning curve on postulate applications.

Combining the first two hypotheses, if we order the problems so that only a few new postulates are introduced for each problem, then we would see a gradual improvement of the students' performances with a little "jump" when new postulates are introduced (Figure 9). Note that the occurrence of postulate applications is sorted within each problem according to the number of practice. As a consequence, we may conclude that AGT is superior to the control tutor in terms of time needed to achieve mastery. The AGT group shows a rapid drop in time for postulate applications within a problem and a small jump at the new problem, whereas the control group shows a loose drop and a big jump at the new problem.

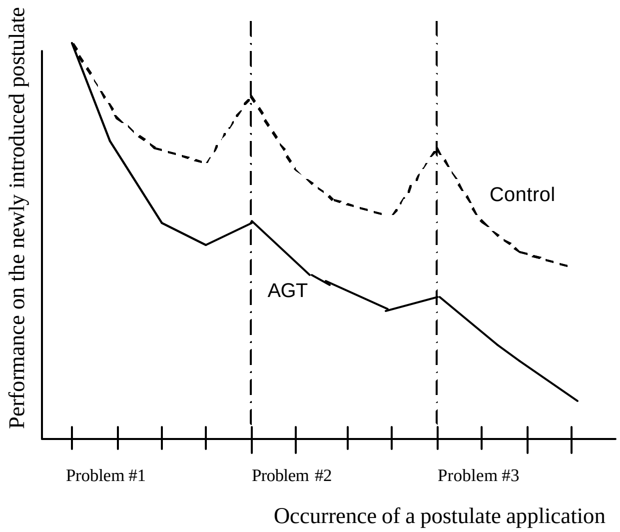

Figure 9: Hypothetical learning curve over the tutoring sessions

Impact on individual differences: Even though we can expect that AGT group would show more speed and more accuracy in learning new postulates than the control group in an aggregated fashion, we have yet to know which tutor can help wider variety of students. To investigate this issue, we compare the number of students that show significant learning gain, which is measured as an aggregated improvement of postulate applications. The larger the number of students who benefit from the tutor, the more robust the tutor is against the students' individual differences. AGT may or may not have more impact on individual difference that the control tutor does.

6 Milestone

Figure 10 shows the schedule of the dissertation work.

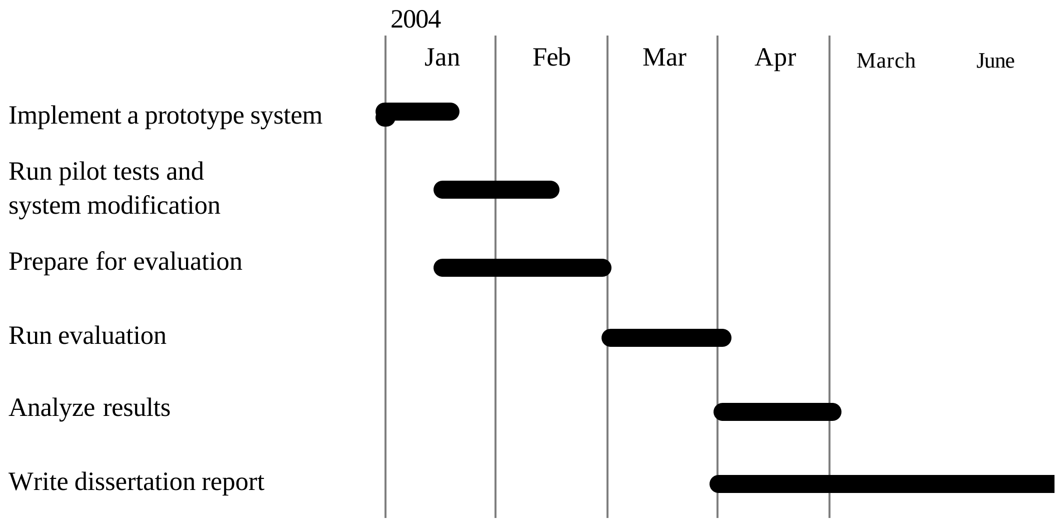

Figure 10: Time line

7 Intellectual Benefits

The proposed study will broadly benefit both engineering of building intelligent tutoring system in problem solving and mathematics education in teaching geometry theorem proving.

Reification of inference steps: It is a novel trial to have a cognitive model that breaks a postulate application into inference steps. This study will provide new knowledge about an impact on having such cognitive model for tutoring. If the results affirmatively support effectiveness of tutoring, then an additional investigation will be desired to see if further elaboration extends the effect. If the results are negative, then we must investigate alternative factors to improve tutoring effects instead of having a labored cognitive model.

Impact of proactive scaffolding: Learning from worked-out examples has been known to be effective when students self-explain the other's work (Chi et al., 1989; Renkl, Stark, Gruber, & Mandl, 1998). We will see if learning from worked-out examples is equally effective when student's self-explanation is replaced with the tutor's explanation that is equally informative as the students' self-explanation. If the results confirm the impact of the tutor's elaborated explanation on student's learning, then the effect of self-explanation in learning is not because that the students *provide* the rationale of a worked-out example, but that they *recognize* it.

An effective tutoring strategy for geometry theorem proving with construction: Geometry theorem proving is a challenging task for students. The difficulty leaps when it involves constructions. No effective tutoring strategy for construction has been reported. Indeed, construction has been considered as a "creative" activity and taught to be solved by heuristics (Polya, 1957). If the results support that Advanced Geometry Tutor is indeed effective, then we will know how to teach theorem proving with construction as a well-structured problem.

A new tool for learning geometry theorem proving: Finally, we hope to provide a learning environment for students to learn proof writing. The software runs on a stereotypical PC, and it can be delivered world wide. Therefore, we will be providing a powerful educational tool for students to learn geometry theorem proving.

Reference:

Anderson, J. R., Bellezza, F. S., & Boyle, C. F. (1993). The Geometry Tutor and Skill Acquisition. In J. R. Anderson (Ed.), *Rules of the mind* (pp. 165-181). Hillsdale, NJ: Erlbaum.

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *Journal of the Learning Sciences*, *4*(2), 167-207.

Chaiyasang, S. (1989). An Investigation into Level of Geometric Thinking and Ability to Construct Proof of Students in Thailand. *Dissertation Abstracts International*, *49A*(8), 2137.

Chi, M. T., Bassok, M., Lewis, M. W., Reimann, P., & Glaser, R. (1989). Self-explanations: How students study and use examples in learning to solve problems. *Cognitive Science*, *13*(2), 145-182.

Greeno, J. G. (1983). Forms of Understanding in Mathematical Problem Solving. In S. G. Paris, G. M. Olson & H. W. Stevenson (Eds.), *Learning and Motivation in the Classroom* (pp. 83-111). Hillsdale, NJ: Erlbaum.

Koedinger, K. R. (1990). *Theoretical and Empirical Motivation for the design of ANGLE: A new geometry learning environment*. Pittsburgh: Carnegie Mellon University.

Matsuda, N., & VanLehn, K. (2004). GRAMY: A Geometry Theorem Prover Capable of Construction. *Journal of Automated Reasoning*.

Polya, G. (1957). *How to Solve It* (2nd ed.). Princeton, NJ: Princeton University Press.

Renkl, A., Stark, R., Gruber, H., & Mandl, H. (1998). Learning from Worked-Out Examples: The effects of example variability and elicited self-explanations. *Contemporary Educational Psychology, 23*(1), 90-108.

Senk, S. L. (1985). How well do students write geometry proofs? *Mathematics Teacher, 78*(6), 448-456.

Sweller, J., & Cooper, G. A. (1985). The use of worked examples as a substitute for problem solving in learning algebra. *Cognition and Instruction, 2*(1), 59-89.

Tubridy, A. F., Jr. (1992). *An Instructional Strategy to Enhance Proof-Writing Ability in Secondary School Geometry.* University of Texas at Arlington.

Usiskin, Z. (1982). *Van Hiele Levels and Achievement in Secondary School Geometry. CDASSG Project* (No. ERIC-ED220288). IL: Chicago University.

VanLehn, K. (1998). Analogy events: How examples are used during problem solving. *Cognitive Science, 22*(3), 347-388.

VanLehn, K., Jones, R. M., & Chi, M. T. (1992). A model of the self-explanation effect. *Journal of the Learning Sciences, 2*(1), 1-59.

Appendix:

A list of geometry postulates taught by Advanced Geometry Tutor. The abbreviation of each postulate is the ones used in the Figure 5 (p.22).

1. CPCTC

Theorem:

Corresponding parts of two congruent triangles are congruent

Operationalization:

Given ΔABC and ΔXYZ, if $\Delta ABC \equiv \Delta XYZ$, then $AB = XY$, $BC = YZ$, $AC = XZ$, $\angle A = \angle X$, $\angle B = \angle Y$, $\angle C = \angle Z$.

2. SelfCong

Theorem:

Every segments and angles are congruent to itself.

Operationalization:

This is a perceptually evident theorem, hence no operationalization required. The students are asked to enter "IDENTITY" as a justification of self congruent propositions.

3. SAS

Theorem:

If two sides and the angle between them in one triangle are congruent to the corresponding parts in the other triangle, then the triangles are congruent.

Operationalization:

Given ΔABC and ΔXYZ, if $AB = XY$, $BC = YZ$, and $\angle A = \angle X$, then $\Delta ABC \equiv \Delta XYZ$

4. VerAng

Theorem:

Vertical angles are congruent.

Operationalization:

Given two segments AB and CD intersecting at P, then $\angle APC = \angle BPD$ and $\angle APD = \angle BPC$

5. Z

Description:

Given two lines cut by a transversal. If the lines are parallel, then a pair of alternative interior angles is congruent.

Operationalization:

Given a line AB and CD, if AB // CD then $\angle ABC = \angle BCD$

6. ASA

Description:

If two angles and the side between them in one triangle are congruent to the corresponding parts in another triangle, then the triangles are congruent.

Operationalization:

Given two triangles ΔABC and ΔXYZ, if $\angle A = \angle X$, $\angle B = \angle Y$, and $AB = XY$, then $\Delta ABC \equiv \Delta XYZ$

7. MTri

Description:

If a segment that intersects with a side of a triangle at its midpoint is parallel to another side, then the segment bisect the third segment.

Operationalization:

Given ΔABC with a point N on AB and M on AC. If $AN = NB$ and $NM \parallel BC$, then $AM = MC$

8. Trans

Description:

If a quantity equals/parallel to two quantities, then these two quantities are equal/parallel.

Operationalization:

If $A = B$ and $A = C$, then $A = C$

If $A \parallel B$ and $A \parallel C$, then $A \parallel C$