Automated Generation of Readable Proofs with Geometric Invariants[†] I. Multiple and Shortest Proof Generation

Shang-Ching Chou, Xiao-Shan Gao[‡] and Jing-Zhong Zhang[§]
Department of Computer Science, The Wichita State University
Wichita KS 67260-0083, USA
e-mail: (chou,gao,zhang)@cs.twsu.edu

Abstract. In this series of papers, we discuss how to use a fixed set of high level geometry lemmas or rules related to geometric invariants, such as area, full-angle, etc., to produce short and human-readable proofs in geometry, especially to produce multiple and shortest proofs of a given geometry theorem. These rules are proved to be much more effective and concise than the rules based on triangle congruence used in related work before. The success of our approach is partially due to a skillful selection of geometric invariants and the related rules. Control and search strategies are proposed and experimented with to enhance the efficiency of the prover. In part I of this series, the high level geometry lemmas are about area and the Ceva-Menelaus configurations.

Keywords. Automated reasoning, automated geometry theorem proving, area method, multiple proof, shortest proof.

1 Introduction

In [3, 4, 20], we introduced the *area method* for automated generation of human-readable proofs of geometry theorems. The computer program based on this method has generated human-readable proofs of more than 400 difficult geometry theorems. By human-readable proofs, we mean that the proofs are short enough for people to repeat with pencil and paper and each step of the proofs has clear geometric meanings.

[†]This work was supported in part by the NSF Grants CCR-9117870, CCR-9420857 and the Chinese NSF.

[‡]On leave from Institute of Systems Sciences, Academia Sinica, Beijing 100080.

[§]Permanent address: Chengdu Institute of Computer Application, Academia Sinica, 610015 Chengdu, China

However, not all proofs generated by the area method are short. We raise the following question: can we use other geometric invariants for automated generation of short and elegant proofs of geometry theorems that our area method is unable to do? By geometric invariants, we mean those geometric quantities which have clear geometric meanings, such as ratio of segments, area, full-angle, etc. This is one of the main themes of this work.

It is our experience with the area method that in order to be successful in generating short proofs, it is crucial to choose the right geometry invariants and the related high level geometry lemmas about these invariants. The success of our approach is partially due to a skillful selection of geometric invariants and the related rules. We are seeking sets of ideal geometric invariants and related methods and hope that they meet the following criteria: (1) The proofs based on the methods should be generally short. Otherwise, we may face the problem of search space explosion before reaching a proof. On the contrary, if the lemmas can be used to produce short proofs efficiently, the computer program may produce a large number of proofs quickly and select the shortest ones among them. This is the basis for multiple proof generation. (2) The methods are powerful enough to prove many difficult geometry theorems without adding auxiliary points or lines. (3) The methods can be used to prove diagram independent proofs for geometry statements. The key here is that the lemmas can deal with the order relation properly. For the detailed discussion of this topic see Section 4.2. (4) The proofs generated by the method should have clear geometric meanings and are easy to read.

Our area method basically satisfies all the above four criteria, i.e., the method can be used to produce short, elegant, and diagram independent proofs for hundreds of difficult geometry theorems within its domain.

In Part I of this series, we extend the area method by adding two new geometry lemmas: Ceva's theorem and Menelaus' theorem. Proofs generated with these two lemmas are unexpectedly beautiful and short. We also propose control and search strategies for the area method to produce multiple and shortest proofs. These strategies will also be used for methods based on other geometric invariants. In Part II of this paper, we introduce another very useful geometry invariant – full-angle.

Why do we search for more methods to mechanize the proving of geometry theorems? First, for a certain class of geometry theorems, a particular method may produce much shorter proofs than other methods. For instance, the area method works particularly well for constructive theorems in affine geometry. On the other hand, the full-angle method (in Part II) works well for theorems involving many circles and angles. Second, with these methods, for the same theorem, the prover can produce a variety of proofs with different styles. This is important for the method to be used in *geometry education*, since different methods may enhance students' ability to explore different and better proofs.

1.1 Multiple and Shortest Proof Generation

Our previous area method is deterministic [3] because the method eliminates points in a rigid way. In this work, we use a relaxed search strategy; thus the new method has some kind of non-deterministism. Using a relaxed search strategy has two positive aspects.

First, using a relaxed search allows the program to generate multiple and shorter proofs for the same theorem. Actually, a new phenomenon we have experienced with the new approach is the *proof explosion*. Our preliminary program produces thousands of proofs even for a simple geometry theorem. Most of the proofs are more or less similar. For instance, the difference of two similar proofs may be caused by interchanging the order of applying two rules. However, among those proofs, we do find a few that are essentially different. For instance, the two different proofs may be produced by applying different sets of lemmas. Control and search strategies are proposed and experimented with to discard most of the similar proofs and keep the essentially different proofs and the shortest proofs. Our precise meaning for the shortest proof can be found in Section 3.3.

Second, by using a relaxed search, we extend our area method in two aspects. (1) The area method we proposed before is only limited to the constructive geometry statements. The approach to the area method in this paper can deal with not only constructive geometry statements [3], but also non-constructive geometry statements that can be described by geometry predicates. (2) Even for the same geometry theorem, our new program can generate shorter proofs that cannot be generated by the previous area method (see Example 3.2), since our new program can find the shortest ones among thousands of proofs.

Most of the previous work of automated theorem proving generally satisfy with one proof. In our case, since the proofs generated are short and readable, generating multiple and shortest proofs for geometry theorems is important for the application of the method/program to *geometry education*. One thing teachers often do in class is asking students to find an alternative or better proof for a geometry theorem. A prover capable of finding multiple and shortest (readable) proofs may help the student to enhance their ability to solve geometry problems.

2 The Prover

2.1 Basic Lemmas About Signed Areas and Ratio of Segments

In this paper, points are represented by capital English letters; lines are represented by two *distinct points* on it, e.g., line AB is the line passing through two distinct points A and B.

The *signed area* S_{ABC} of triangle ABC is the usual area with a sign depending on the order of the vertices A, B, and C: if $A-B-C$ rotates counterclockwisely, S_{ABC} is positive, otherwise it is negative. In our area method, the signed area is used as a basic (undefined) geometry quantity and its properties are used as axioms for our deduction. The following properties of the signed area are used as basic lemmas.

L1 $S_{ABC} = S_{CAB} = S_{BCA} = -S_{BAC} = -S_{CBA} = -S_{ACB}$.

L2 Points A, B, and C are collinear iff $S_{ABC} = 0$.

L3 $PQ \parallel AB$ iff $S_{PAB} = S_{QAB}$.

L4 For any four points A, B, C, and D, we have $S_{ABC} = S_{ABD} + S_{ADC} + S_{DBC}$.

L5 If points A, B, C, and D are four collinear points such that $A \neq B$, and P is any point not on line AB, then $\frac{\overline{CD}}{\overline{AB}} = \frac{S_{PCD}}{S_{PAB}}$.

Here we introduce another geometry quantity $\frac{\overline{CD}}{\overline{AB}}$, the ratio of two directed segments on the same line, which satisfies L8 below.

The *signed area of a quadrilateral* $ABCD$ is defined to be $S_{ABCD} = S_{ABC} + S_{ACD}$. By lemmas L4 and L1, we have.

Q1 $S_{ABCD} = S_{ABC} + S_{ACD} = S_{ABD} - S_{CBD}$,

Q2 $S_{ABCD} = S_{BCDA} = S_{CDAB} = S_{DABC} = -S_{ADCB} = -S_{DCBA} = -S_{CBAD} = -S_{BADC}$.

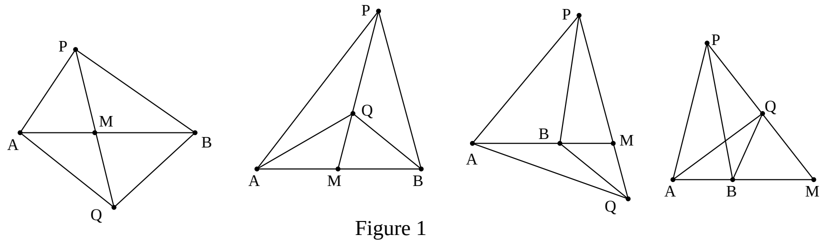


Figure 1

L6 (The Co-side Theorem) *Let M be the intersection of two lines AB and PQ and $Q \neq M$. Then*

$$\frac{\overline{PM}}{\overline{QM}} = \frac{S_{PAB}}{S_{QAB}}; \ \frac{\overline{PM}}{\overline{PQ}} = \frac{S_{PAB}}{S_{PAQB}}; \ \frac{\overline{QM}}{\overline{PQ}} = \frac{S_{QAB}}{S_{PAQB}}.$$

Note that the three equations are valid in all of the four cases in Figure 1.

L7 *Let R be a point on line PQ. Then for any two points A and B,* $S_{RAB} = \frac{\overline{PR}}{\overline{PQ}} S_{QAB} + \frac{\overline{RQ}}{\overline{PQ}} S_{PAB}$.

L8 For four distinct collinear points P, Q, A, and B, $\frac{\overline{PQ}}{\overline{AB}}$ is a real number which satisfies (1) $\frac{\overline{PQ}}{\overline{AB}} = -\frac{\overline{QP}}{\overline{AB}} = \frac{\overline{QP}}{\overline{BA}} = -\frac{\overline{PQ}}{\overline{BA}}$; (2) $\frac{\overline{PQ}}{\overline{AB}} \cdot \frac{\overline{AB}}{\overline{PQ}} = 1$; (3) $\frac{\overline{PQ}}{\overline{AB}} = 0$ iff $P = Q$; and (4) $\frac{\overline{AP}}{\overline{AB}} + \frac{\overline{PB}}{\overline{AB}} = 1$.

These lemmas are a part of the geometry lemmas used in the area method which can be found in [3]. The difference between our previous work [3] and this work is that here we add control and search strategies to cover a wider range of geometry theorems and to produce multiple and shortest proofs. New lemmas about signed area will be introduced in Subsection 2.5. Also, lemmas about another geometric invariant, the Pythagorean difference, is omitted.

In our Prolog program, each geometry lemma is represented as a *rule* or a *clause*. For instance, lemma L3 can be written as the following Prolog rule.

el_rule([area,P,A,B],[[area,Q,A,B],1]) :- para(A,B,P,Q),P≠Q,

where [area,P,A,B] is S_{PAB} and the predicate para(A,B,P,Q) means that $AB \parallel PQ$. If [area,P,A,B] is the input, predicate para is used to find a new point Q such that $AB \parallel PQ$, and hence $S_{PAB} = S_{QAB}$.

Generally speaking, a rule in our program has the following form:

el_rule(Inv,[N, D]) :- $P_1, \cdots, P_t$.

where Inv is a geometry invariant, N and D are polynomials of geometry invariants, and P_i are geometry predicates. The meaning of the rule is that when certain points satisfy the predicates P_i, the invariant Inv will be replaced by the rational expression $\frac{N}{D}$. To find points satisfying the predicates P_i, we only use the hypotheses of the geometry statement. In Part II of the series, we will discuss how to extend the original hypotheses to a geometry information base for the purpose of finding points to satisfy predicates P_i in the rule. The reason that the area method can work with the original hypotheses is that it needs only very weak "local" facts.

Remark 2.1 1. In lemmas L5, L6, L7, and L8, we introduce some conditions like two points are not equal or three points are not collinear. These conditions are usually not in the original hypotheses of the geometry statement. We call them the *non-degenerate (ndg) conditions* for the statement [2, 18]. The machine proof (and generally the statement itself) is valid only under these non-degenerate conditions. Algebraically, these conditions are used to make denominators of the algebraic expressions in the proof not vanishing. In our program, the ndg conditions are treated with the *negation by failure* strategy used in Prolog.

2. Note that some of the lemmas can be used in several ways. For instance, the equation $\frac{\overline{PM}}{\overline{QM}} = \frac{S_{PAB}}{S_{QAB}}$ in L6 can also be used as $S_{PAB} = \frac{\overline{PM}}{\overline{QM}} S_{QAB}$.

2.2 The Prover

The prover is implemented in Prolog. Ideally, we can put the geometry lemmas as inference rules in Prolog, and leave the logic engine of Prolog to do as much inference job as possible. The advantage of doing this is that the prover becomes extensible. If we want to use new lemmas, we can easily add them as inference rules. Since we do not use function symbols, unification without occur check is not a problem in our Prolog program. At the highest level, our prover uses a typical *backward chaining* search strategy.

Step1 The prover first transforms the conclusion predicate into an equation of geometric quantities $\alpha = \beta$.

Step2 The prover keeps replacing geometry quantities in α and β with expressions in new geometry quantities by using the rules introduced in the preceding subsection until no new replacement can be made. After applying a rule to α or β to obtain α' and β', we remove the common factors of α' and β'.

Step3 Let the final equation be $\alpha'' = \beta''$. If α'' literally equals to β'' then the geometry is true. Otherwise, in some special cases, e.g., if α'' and β'' are nonzero-constants or an expression involving free points in the case of constructive statements, the statement is wrong. In the general case, we do not know whether the statement is true or not.

Step4 The prover automatically generates the *complete machine proof*, if needed, in TeX typesetting form.

The termination of the algorithm is based on Control Strategy 1 which will be introduced in the next subsection. It is clear that the prover also uses the computation of symbolic polynomials of geometry quantities, which are builtin in our program.

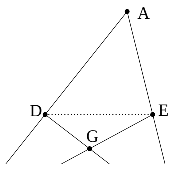
In our previous work on the area method [3], the rules introduced in Subsection 2.1 correspond to the basic propositions in [3], and the elimination lemmas used in [3] can be obtained as combination of several of the basic rules. Therefore, the method we adopt here includes the method in [3] as a special case; thus the method is complete (decidable) for constructive geometry statements.

2.3 An Example

We limit ourselves to geometry statements of the form

$$\forall points(Hyp \Rightarrow Conc)$$

where *Hyp* is a set of predicates which consists of the hypothesis and *Conc* is a predicate or an equation of geometry invariants which is the conclusion. The predicates used in the program are: coll (collinear), para (parallel), perp (perpendicular), cong (segment congruent), eqangle (full-angle congruent; see Part II), midpoint, cyclic. foot, circumcenter, orthocenter, incenter, pbisector (perp-bisector).



Example 2.2 (The Centroid Theorem)
The three medians of a triangle meet in a point, and each median is trisected by this point.

Point order: A, B, C, D, E, G.

Hypotheses: midpoint(E, A, B), midpoint(D, A, C), coll(G, B, D), coll(G, C, E).

Conclusion: $-2 \cdot \frac{\overline{GD}}{\overline{GB}} = 1$.

Non-degenerate condition: A, B, C are not collinear.

The non-degenerate (ndg) condition is generated by our method automatically.

The Machine Proof

$(-2) \cdot \frac{\overline{GD}}{\overline{GB}}$

($\frac{\overline{GD}}{\overline{GB}} = \frac{S_{EDC}}{-S_{ECB}}$, because collinear($B$,$D$,$G$), collinear($C$,$E$,$G$). (L6))

$= \frac{2 \cdot S_{EDC}}{S_{ECB}}$

($S_{EDC} = \frac{1}{-2}(S_{ECA})$, because midpoint($D$,$C$,$A$). (L5))

($S_{ECB} = \frac{1}{2}(S_{CBA})$, because midpoint($E$,$B$,$A$). (L5))

$= \frac{2 \cdot S_{ECA}}{(-1) \cdot S_{CBA}}$

($S_{ECA} = \frac{1}{-2}(S_{CBA})$, because midpoint($E$,$B$,$A$). (L5))

$= \frac{S_{CBA}}{S_{CBA}}$

(Simplification: removing the factor: S_{CBA}.)

$= 1$

Ndg conditions: B,A,C are not collinear.

The machine proof format is a series of *successive equations* of rational expressions in geometry invariants and each equation is obtained by replacing some geometry invariants with expressions in new invariants. The geometry conditions and the name of the rules used in each replacement are also given.

2.4 A Control Strategy Related to Point Order

Control Strategy 1. We introduce a rank among all geometry invariants. A rule can be used only if it reduces an invariant to an expression in invariants with lower ranks.

The rank for the area and ratio of segments is determined by an order among the points. Suppose that we have an order among the points, denoted by $<$. First, the area and ratio of segments can be represented canonically.

Definition 2.3 S_{ABC} is said to be in the *canonical form* if $A \geq B \geq C$. S_{ABCD} is said to be in *canonical form* if $A \geq C$, $B \geq D$, and ($A > B$ or ($A = B$ and $C > D$)). $\frac{\overline{AB}}{\overline{CD}}$ is said to be in the *canonical form* if $A \geq B$, $C \geq D$, and ($A > C$ or ($A = C$ and $B > D$)).

Using rules L1, Q2, and L8, any geometry quantity can be reduced to canonical form. In what follows, all geometry quantities are assumed to be in canonical form. Also in the prover, geometry quantities are represented in canonical forms to enhance search efficiency.

Definition 2.4 A geometry quantity G_1 has lower rank than G_2, denoted by $G_1 < G_2$, if one of the following conditions is true.

1. $G_1 = S_{ABC}$, $G_2 = S_{PQR}$, and the ordered point set $\{A, B, C\}$ has a lower rank than $\{P, Q, R\}$ in the pure lexicographical order associated with the point order;
2. $G_1 = \frac{\overline{AB}}{\overline{CD}}$, $G_2 = \frac{\overline{PQ}}{\overline{RT}}$, and $\{A, B, C, D\}$ has a lower rank than $\{P, Q, R, T\}$ in the pure lexicographical order associated with the point order;
3. $G_1 = \frac{\overline{AB}}{\overline{CD}}$, $G_2 = S_{PQR}$, and $A \leq P$.

Control Strategy 1 has a two-fold purpose. First, it guarantees the termination, because in each step a geometry quantity will be replaced by another quantity of lower rank and there is no infinite chain of geometry quantities with decreasing rank. Second, it makes the production of short proofs possible. Without this strategy, the prover may do endless searches before reaching the shortest proof. This can be seen from Example 3.1.

It is clear that Control Strategy 1 generally will reduce the number of applicable rules. So it is natural to ask: does this strategy reduce the power of the prover? The answer is generally not. According to our experience of proving hundreds of geometry theorems with computer programs, most of the geometry theorems (not involving inequalities) can be described constructively, i.e., points are introduced by successfully taking the intersections of lines and circles. Thus there is a natural order among the points in the statement, i.e., the order according to which the constructed points are introduced. For such a constructive theorem, at least one proof in accordance with Control Strategy 1 exists, i.e., the proof produced by our previous area method [3] which is complete (decidable) for the class of constructive statements.

2.5 More Lemmas

To enhance the efficiency of the prover, we actually builtin more sophisticated rules (lemmas) into our prover. These rules are usually combinations of several basic rules. There are two kinds of combined rules. The first kind is to eliminate a point from a geometry quantity. This kind of combined rules corresponds to the elimination lemmas in [3, 4]. The second kind is chosen based on our observation that these combinations are often used in the proofs of many theorems. Theoretically speaking, these combined rules can be automatically reached by searching among the basic rules. But the searching process is time consuming, since the prover has to go through many "bad" combinations before obtaining a "good" one. We give some of the combined rules as illustrations.

L9 If Y is the intersection of line AB and line EF, then for any point C we have $S_{YAC} = \frac{\overline{YA}}{\overline{BA}} S_{BAC} = \frac{S_{AEF} S_{BAC}}{S_{AEBF}}$.

L10 If Y, B, and C are collinear and $YA \parallel BD$, $S_{YCD} = S_{YBD} + S_{BCD} = S_{ABD} + S_{BCD}$

L11 Let the diameter for the circumcircle of triangle ABC be δ. Then $S_{ABC} = \frac{\overline{AB} \cdot \overline{BC} \cdot \overline{CA}}{2\delta}$.

Rule L9 is a combination of rules L5 and L6; rule L10 is a combination of rules L4 and L3. The proof for L11 may be found in [3].

A new kind of rules used in the prover is *match*. The rules introduced previously apply only to one geometry quantity. The match rules on the other hand can apply to more than one geometry quantities. The reason to introduce the match rules is to enhance the efficiency and to obtain shorter proofs.

L12 If A, B, and C are collinear then $S_{PAB} + S_{PBC} = S_{PAC}$.

L13 If O, A, B are collinear and O, C, D are collinear then $S_{OBC} + S_{ODA} = S_{ABCD}$.

L14 $S_{ABD} - S_{CBD} = S_{ABCD}$.

L15 If points A, B, C, and D are four collinear points and P is any point not on line AB, then $\frac{S_{PCD}}{S_{PAB}} = \frac{\overline{CD}}{\overline{AB}}$.

L16 Let M be the intersection of two non-parallel lines AB and PQ. Then $\frac{S_{PAB}}{S_{QAB}} = \frac{\overline{PM}}{\overline{QM}}$.

Rules L12 and L13 are consequences of rule L4. Rule L14 is from the definition of the area of quadrilateral. Rule L15 is a consequence of rule L5. Rule L16 is a consequence of rule L6.

Example 2.5 Continuing from Example 2.2. The following is another machine produced proof for the centroid theorem. Notice that in the third step, it uses a match rule L15.

The Machine Proof

$(-2) \cdot \frac{\overline{GD}}{\overline{GB}}$

 $(\frac{\overline{GD}}{\overline{GB}} = \frac{S_{EDC}}{-S_{ECB}}$, because collinear($B$,$D$,$G$), collinear($C$,$E$,$G$). (L6))

$= \frac{2 \cdot S_{EDC}}{S_{ECB}}$

 $(S_{EDC} = \frac{1}{-2}(S_{ECA})$, because midpoint($D$,$C$,$A$). (L5))

 $(S_{ECB} = \frac{1}{2}(S_{CBA})$, because midpoint($E$,$B$,$A$). (L5))

$= \frac{2 \cdot S_{ECA}}{(-1) \cdot S_{CBA}}$

 $(\frac{S_{ECA}}{S_{CBA}} = -\frac{\overline{EA}}{\overline{BA}}$, because collinear($A$,$B$,$E$). (L15))

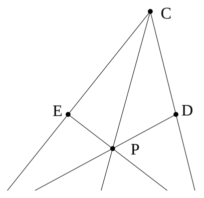
$= 2 \cdot \frac{\overline{EA}}{\overline{BA}}$

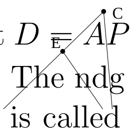
 $(\frac{\overline{EA}}{\overline{BA}} = \frac{1}{2}$, because midpoint($E$,$B$,$A$). (L8))

$= 1$

Ndg conditions: B,A,C are not collinear.

Finally, we add Ceva's theorem and Menelaus' theorem as the basic rules of proving geometry theorems. These two theorems have been used to produce elegant proofs for many difficult geometry theorems. The proofs are unexpectedly beautiful and short. Two of them are in the appendix (Examples 4 and 5). Here, we mention again that Control Strategy 1 is important for the prover to find the proof efficiently.





LC (Ceva's Theorem) For four points A, B, C, and P, let $D = AP \cap CB$, $E = BP \cap AC$, and $F = CP \cap AB$ (Figure 3). Then $\frac{\overline{AF}}{\overline{FB}} = \frac{\overline{DC}}{\overline{BD}} \cdot \frac{\overline{EA}}{\overline{CE}}$. The ndg condition is that P is not on the lines AB, AC, and BC. Triangle DEF is called the *cevain triangle* of point P for triangle ABC.

LM (Menelaus' Theorem) A transversal meets the three sides AB, BC, and CA of a triangle ABC in F, D, and E respectively (Figure 4). Then $\frac{\overline{AF}}{\overline{FB}} = -\frac{\overline{DC}}{\overline{BD}} \cdot \frac{\overline{EA}}{\overline{CE}}$. The ndg condition is that A, B, and C are not on line EF.

3 Multiple and Shortest Proof Generation

The area method in [3] is very effective: it can generate a proof for 85 percent of the 478 theorems within one second [3]. This makes it possible for us to generate many proofs for the same geometry theorem efficiently.

3.1 Algebraic Computation Versus Synthetic Deduction

Wen-Tsun Wu has shown that algebraic computation is an efficient way of proving difficult geometry theorems [17, 18]. On the contrary, in the synthetic approaches, rarely is algebraic computation used. In [10], it is reported that algebraic computation may cause explosion of the search space, and hence should be avoided.

In our opinion, certain kind of algebraic computation is necessary for a powerful geometry theorem prover. One reason is that algebraic quantities can be manipulated efficiently in computers. Another reason is that even the traditional geometry uses simple algebraic computations in solving of moderately difficult geometry problems. For example, in Euclid's *Elements*, the theory of ratios is introduced before the discussion of similar triangles. We believe that the right way of automatically producing readable proofs for geometry theorems should be a combination of the algebraic computation with the synthetic deduction: using simple algebraic operations to achieve the efficiency and using synthetic deduction to preserve the elegance of the traditional geometry proofs. Our area method is such a combination.

In our approach based on geometric invariants, a proof is written as a series of *successive equations* of algebraic expressions in geometry invariants and each algebraic expression has the form

$$\frac{P_1 \cdot P_2 \cdots P_n}{R_1 \cdot R_2 \cdots R_m}$$

where P_i and R_i are polynomials in geometry invariants. We do not expand the products of the polynomials, because after using each rule it is often the case that some polynomials occur in both the numerator and the denominator and hence can be canceled. This is one of the reasons that our program may produce short proofs.

3.2 Multiple Proof Generation

We use two approaches to generate different proofs. First, for an algebraic quantity

$$\alpha = \frac{P_1 \cdot P_2 \cdots P_n}{R_1 \cdot R_2 \cdots R_m}$$

we can use the rules to one or several of the polynomials in different orders. Even by simply changing the order of applying the same set of rules to the algebraic quantities, essentially different proofs may be generated, because when some polynomial, say P_1, becomes Q_1 and R_1 is unchanged, it might happen that $Q_1 = R_1$ could be canceled. If we change R_1 to S_1 at the same time, the above reduction will not happen. See Example 3.2. In our prover, we use the following strategy to control the order of elimination, which allows us to reach all combinations of different eliminations.

Control Strategy 2. To apply a set of rules to an algebraic expression

$$\alpha = \frac{P_1 \cdot P_2 \cdots P_n}{R_1 \cdot R_2 \cdots R_m}$$

means either to apply one rule to one of the polynomials $P_1, \cdots, P_n, R_1, \cdots, R_m$, or to apply one rule to one of the polynomials in the numerator and to apply another rule to one of the polynomials in the denominator.

The second way of generating different proofs is more important. For each geometry invariant, we may apply different rules to it to obtain different results. Using this strategy, many essentially different proofs may be generated.

Besides Strategies 1 and 2, other efforts are made to reduce the production of too many similar deductions.

Control Strategy 3. If two rules are used to eliminate the same point from both the numerator and the denominator of an algebraic expression or used to change the types of the geometry quantities, we will not use them separately to the same algebraic expression, since the separate use of these rules will not lead to canceling of geometry quantities.

If not using Strategies 1, 2, and 3, even for a simple geometry theorem as the centroid theorem, tens of thousands of proofs can be generated. We call this phenomenon *proof explosion*.

Example 3.1 Continuing from Example 2.2.

If all the three control strategies (1, 2, and 3) are used, the prover gives 70 proofs with proof lengths ranging from 4 to 7. Three of the proofs can be found in Examples 2.2, 2.5, and 3.2.

If we relax the control by dropping control strategy 3, then the prover gives 212 proofs with lengths ranging from 4 to 8.

If we further relax the control by dropping the control strategy 1, then the prover gives 24360 proofs before it exhausts the memory of the computer. Much more proofs are expected, because in the 24360 proofs there are only two proofs with length less than or equal to 8, i.e., only two of the 215 proofs obtained by adopting strategies 1 and 2 occur in the 24360 proofs. Therefore, strategies 1,2, and 3 are quite effective to control the number and the length of the proofs.

3.3 Search Strategies and Shortest Proof Generation

The default search strategy for the prover is *depth-first.* The advantage of using the depth-first search is that we can obtain the first proof for a statement very fast (see Subsection 4.3). Since we are only interested in short proofs, the program allows the user to use a *bounded depth-first* search to find the proofs with lengths less than a fixed number.

A related topic is the finding of *shortest* proofs for a geometry statement under certain rules. If using the *breadth-first search*, the first available proof is naturally a shortest proof. But if using the breadth-first search, for some difficult geometry problems, the prover may exhaust the computer space before it obtains the first (shortest) proof. We solve this problem by using the following natural technique: if the shortest proof is needed, the prover will remember the length of a proof when it is generated and prohibit any backtracking that can generate proofs with lengths longer than or equal to the recorded length. Therefore, the prover will produce proofs with lengths decrease strictly and obtain the shortest proof fast. For some really difficult theorems, this method can at least give the shortest possible proofs within the limit of the computer time and space.

We also implement another approach: the *depth-first iterative deepening search* [11, 14]. In this approach, we first try to find a proof with depth 1, then depth 2, and so on by repeated use of depth-first search. Korf [11] has shown that this approach is asymptotically optimal among brute-force search strategies in terms of solution length, space, and time. It also has the property that the first proof found is a shortest proof.

Our experience shows that both the above approaches to find shortest proofs are quite successful: using the depth-first search, our prover fails to find all the proofs for some examples after running 10 hours; but the prover does find the shortest proofs within reasonable time using both approaches to find the shortest proof. See Section 6 in Part II.

It is clear that the shortest proof generated by the above method is only the shortest one relative to the control strategies used by us. An absolute shortest proof can be obtained only if we drop all the control strategies which prevent the usage of certain rules. But it is our belief that the control strategies adopted by us are reasonable and the shortest proofs relative to these strategies are generally the absolute shortest ones. This is partially supported by the data given in Example 3.1.

Example 3.2 Continuing from Example 3.1. The following proof for the centroid theorem is the shortest proof given by our prover. Note that in the proof, point E is not eliminated by rules. Actually it is eliminated “accidentally” by deleting a common factor S_{ECA} in the last step of the proof. This proof can not be generated by our previous area method which eliminates a point at each step.

The Machine Proof

$(-2) \cdot \frac{\overline{GD}}{\overline{GB}}$

($\frac{\overline{GD}}{\overline{GB}} = \frac{S_{EDC}}{-S_{ECB}}$, because collinear($B$,$D$,$G$), collinear($C$,$E$,$G$). (L6))

$= \frac{2 \cdot S_{EDC}}{S_{ECB}}$

($S_{EDC} = \frac{1}{-2}(S_{ECA})$, because midpoint($D$,$C$,$A$). (L5))

($S_{ECB} = -S_{ECA}$, because midpoint(E,B,A). (L5))

$= \frac{S_{ECA}}{S_{ECA}}$

(Simplification: removing the factor: S_{ECA}.)

$= 1$

Ndg conditions: B,A,C are not collinear.

4 Final Comments

4.1 Related Work

Producing readable proofs for geometry theorems goes back to as early as the late 50s in the work [6, 7] by H. Gelertner, J.R. Hanson, and D.W. Loveland. The main tool in this and the following work ([9, 12, 1, 5]) is the lemmas of triangle congruence. In our opinion, while the technique of congruent triangles is very useful for proving many theorems of high school level, it seems unable to prove a large number of moderately difficult theorems for three reasons. First, the proofs for most of the moderate difficult geometry theorems use tools other than the triangle congruence. Second, the triangle congruence techniques are order or diagram related and there is still no efficient and strict way of dealing with order problems. Third, even in those proofs based on triangle congruence, auxiliary points or lines are often needed to form the required congruent triangles. As we know, finding auxiliary lines and points is one of the most difficult steps in solving geometry problems, and there is still no effective method for doing this automatically. A. Robinson suggested that the auxiliary points and lines can be "constructed" as elements of the Herbrand universe for the problem [13]. But no implementation based on this approach has been given.

While the aim of some of the earlier work was more concerned with the means (a thorough understanding of the organization of information processing activity in the theorem proving and mathematical discovery) than with the end, the aim of our work is mainly concerned with generation of short and elegant proofs of much more difficult geometry theorems.

Our approach here is quite different from the *algebraic methods* which were based on some general algebraic theories, mainly the characteristic set method and the Gröbner basis method [17, 2, 8]. These algebraic methods have been used to prove astonishing difficult geometry theorems. In essence, the above algebraic methods can be characterized as to "trade qualitative difficulty for quantitative complexity" (words of H. Wang [15]), that is the proofs are carried out by massive algebraic computation. Proofs produced by these methods generally lose the elegant manner of the traditional geometry proofs. The basic geometry quantity used in these algebraic methods is the coordinates of points, whereas the basic quantities used in our method are geometric invariants, such as area, ratio of two segments etc., which have clear geometric meanings. Also the proofs produced by the new methods are generally shorter.

4.2 Produce Diagram Independent Proofs

The use of the numerical diagram as the semantic model has been the cornerstone of most of the previous efforts to mechanize plane geometry theorem proving [7, 9, 1, 5]. There are two benefits the early provers derive from a numerical diagram. (1) The diagram is used as a filter to reject goals not consistent with its numerical representation. In other words, the diagram is used as a counterexample. (2) More importantly, the numerical diagram is used to determine order relations among points, or points and lines. These relations are necessary for the prover to find a proof.

While as a counterexample the diagram is used to control the search space successfully, the second benefit of determining order relation has some theoretical problems. Since only one or several numerical examples are checked, the previous provers have the risk of proving only some special cases of the theorem. Nevins [12] claimed that he has got rid of this drawback by adding the ordering relations to the hypotheses of the statement. Nevins' approach makes the situation much clear, but still does not solve the problem completely. First, to prepare for the order relation in the hypotheses, people still need to consult a diagram. Second, for some geometry theorems it may happen that the order of points in *different diagrams of the same theorem* may be different. For instance, there are at least three diagrams with different order relations among points and lines for Ceva's theorem (Figure 5) and the Butterfly theorem (C, D, E, F are four points on a circle. $G = CF \cap DE$, $OG \perp MN$. Show that $MG = NG$. Figure 6). Thus, a proof based on a fixed order relation among points and lines is valid only for some special cases of the statement.

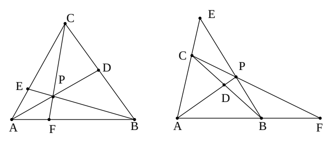


Figure 5

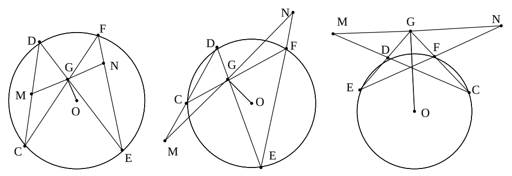


Figure 6

The proofs produced by the area method and the method based on full-angles (in Part II) are independent of the diagrams. For instance, the proofs for Ceva's theorem and the butterfly theorem (Examples 1 and 3 in the appendix) are valid for all diagrams. Also see Example 2.1 in Part II of this series.

A key fact behind the success of our method is that the validity of most elementary geometry theorems involving equalities only is independent of the relative order positions of the points involved. Such geometry theorems belong to *unordered geometry.* This idea is originated from Wu's algebraic method of automated reasoning [18]. In unordered geometry, the proofs of these theorems can be very simple. However, the ordinary proofs of these theorems involve the order relation (among points and lines), hence are not only complicated, but also not strict. The area method is actually for the unordered geometry. For instance, rule L6 is valid for all the four possible diagrams (Figure 1) with different order relations among the points.

By emphasizing diagram independent proofs, we do not diminish the importance of the diagram in geometry theorem proving. By diagram independent proofs, we mean that we can not use conditions coming from the numerical diagram. There are two kinds of order relations. First, the order among some points are invariant. For instance, in Figure 6 point G is always between points M and N. In this case, a proof for this order relation is needed. Second, the order relation among points may change. For instance, in Figure 6 point M may be in or outside segment CD. In this case, to obtain a diagram independent proof, one either does not use this order relation or gives different proofs in different cases. The machine proofs produced by the area method do not depend on these order relations and thus are valid for all diagrams.

4.3 Comparison Between Different Search Strategies

The prover is implemented using the SB-Prolog [16] on a SPARC-10 Workstation. The following table contains some statistics for Example 2.2, and the examples in the Appendix. We include six indexes for each theorem. Steps is the length or steps of the shortest proof obtained. Maxt is the length of the maximal algebraic formula in the shortest proof. Ftime is the time needed to obtain the first proof using the depth-first search. Mtime is the time needed to obtain the shortest proof using the depth-first search. Btime is the time needed to obtain the first proof using the depth-first iterative deepening search. Allprs is the number of all the proofs generated using depth-first search.

Examples	steps	maxt	ftime (secs)	mtime (secs)	btime (secs)	allprs
2.2	4	1	0.79	1.87	2.37	70
1	5	1	0.61	1.42	2.72	2
2	4	2	0.51	5.68	1.7	168
3	11	1	0.66	1.65	24.63	4
4	3	1	0.85	3.47	0.69	16
5	4	1	0.57	2.06	1.25	16

References

[1] J. R. Anderson, C. F. Boyle, A. Corbett, & M. Lewis, The Geometry Tutor, in *Proc. of the IJCAI*, Los Angeles, USA, 1985, p. 1–7.

[2] S. C. Chou, *Mechanical Geometry Theorem Proving*, D.Reidel Publishing Company, Dordrecht, Netherlands, 1988.

[3] S. C. Chou, X. S. Gao, & J. Z. Zhang, *Machine Proofs in Geometry*, World Scientific, 1994.

[4] S. C. Chou, X. S. Gao, & J. Z. Zhang, Automated Production of Traditional Proofs for Constructive Geometry Theorems, *Proc. of Eighth IEEE Symposium on Logic in Computer Science*, p.48–56, IEEE Computer Society Press, 1993.

[5] H. Coelho & L. M. Pereira, Automated Reasoning in Geometry Theorem Proving with Prolog, *J. of Automated Reasoning*, vol. 2, p. 329-390, 1986.

[6] H. Gelertner, Realization of a Geometry-Theorem Proving Machine, *Computers and Thought*, eds. E.A. Feigenbaum, J. Feldman, p. 134-152, Mcgraw Hill.

[7] H. Gelertner, J.R. Hanson, and D.W. Loveland, Empirical Explorations of the Geometry-theorem Proving Machine, *Proc. West. Joint Computer Conf.*, 143-147, 1960.

[8] D. Kapur, Geometry Theorem Proving Using Hilbert's Nullstellensatz, *Proc. of SYMSAC'86*, Waterloo, 1986, 202–208.

[9] P. C. Gilmore, An Examination of the Geometry Theorem Proving Machine, *Artificial Intelligence*, 1, p. 171–187.

[10] K. R. Koedinger and J. R. Anderson, Abstract Planning and Perceptual Chunks: Elements of Expertise in Geometry, *Cognitive Science*, 14, 511-550, (1990).

[11] R. E. Korf, Depth-first Interactive-deepening: an Optimal Admissible Tree Search, *Artificial Intelligence*, 27, 1, p.97-109, 1985.

[12] A.J. Nevins, Plane Geometry Theorem Proving Using Forward Chaining, *Artificial Intelligence*, 6, 1-23.

[13] A. Robinson, Proving a Theorem (as done by Man, Logician, or machine), in *Automation of Reasoning*, ed. by J. Siekmann and G. Wrightson, p.74-78, 1983, Springer-Verlag.

[14] M. Stickel, A Prolog Technology Theorem Prover: Implementation by an Extended Prolog Compiler, *J. of Automated Reasoning*, 4(4), 353–380, 1985.

[15] H. Wang, A Variant to Turing's Theory of Computing Machines, *J. of ACM*, vol. 4, 1957, p.63–92.

[16] D. S. Warren, F. Pereira, & S. K. Debray, *The SB-Prolog System, Version 3.0* Department of Computer Science, Univ. of Arizona.

[17] Wu Wen-tsün, On the Decision Problem and the Mechanization of Theorem in Elementary Geometry, *Scientia Sinica* 21(1978), 159–172; Also in *Automated Theorem Proving: After 25 years*, A.M.S., Contemporary Mathematics, 29(1984), 213–234.

[18] Wu Wen-tsün, *Basic Principles of Mechanical Theorem Proving in Geometries*, Volume I: Part of Elementary Geometries, Science Press, Beijing (in Chinese), 1984.

[19] J. Z. Zhang & P. S. Cao, *From Education of Mathematics to Mathematics for Education,* Sichuan Educational Publishing Inc., (in Chinese) 1988.

[20] J. Z. Zhang, S. C. Chou, & X. S. Gao, Automated Production of Traditional Proofs for Theorems in Euclidean Geometry, I. The Hilbert Intersection Point Theorems, TR-92-3, Department of Computer Science, WSU, 1992. (to appear in *Annals of Mathematics and Artificial Intelligence*)

Appendix. Some Examples and Their Machine Produced Proofs

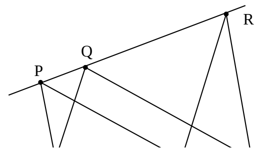
Example 1 (Ceva's Theorem) For four points A, B, C, and P, let $D = AP \cap CB$, $E = BP \cap AC$, and $F = CP \cap AB$ (Figure 3). Then $\frac{\overline{AF}}{\overline{FB}} \cdot \frac{\overline{BD}}{\overline{DC}} \cdot \frac{\overline{CE}}{\overline{EA}} = 1$.

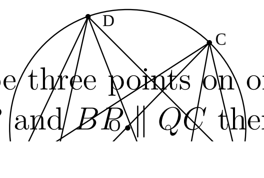
Point order: A, B, C, P, D, E, F.
Hypotheses: coll(A, F, B), coll(C, P, F), coll(B, D, C), coll(A, P, D), coll(C, E, A), coll(B, P, E).
Conclusion: $\frac{\overline{AF}}{\overline{FB}} \cdot \frac{\overline{BD}}{\overline{DC}} \cdot \frac{\overline{CE}}{\overline{EA}} = 1$.

The Machine Proof
$-\frac{\overline{EC}}{\overline{EA}} / \frac{\overline{FB}}{\overline{FA}} \cdot \frac{\overline{DC}}{\overline{DB}}$
($\frac{\overline{EC}}{\overline{EA}} = \frac{S_{PCB}}{-S_{PBA}}$, because collinear($A$,$C$,$E$), collinear($B$,$E$,$P$). (L6))
($\frac{\overline{FB}}{\overline{FA}} = \frac{S_{PCB}}{S_{PCA}}$, because collinear($A$,$B$,$F$), collinear($C$,$F$,$P$). (L6))
$= \frac{S_{PCB} \cdot S_{PCA}}{\frac{\overline{DC}}{\overline{DB}} \cdot S_{PCB} \cdot S_{PBA}}$
(Simplification: removing the factor: S_{PCB}.)
$= \frac{S_{PCA}}{\frac{\overline{DC}}{\overline{DB}} \cdot S_{PBA}}$
($\frac{\overline{DC}}{\overline{DB}} = \frac{S_{PCA}}{S_{PBA}}$, because collinear($B$,$C$,$D$), collinear($A$,$D$,$P$). (L6))
$= \frac{S_{PCA} \cdot S_{PBA}}{S_{PCA} \cdot S_{PBA}}$
(Simplification: removing the factors: S_{PCA}, S_{PBA}.)
$= 1$

Ndg conditions: B,P,A are not collinear; C,P,A are not collinear; A,P,B are not collinear.

Note that the first algebraic formula in the proof, $-\frac{\overline{EC}}{\overline{EA}} / \frac{\overline{FB}}{\overline{FA}} \cdot \frac{\overline{DC}}{\overline{DB}}$, is different from the input. The reason is that in the input, the user can write the geometry quantities arbitrarily, while in the machine proof all geometry quantities are automatically transformed into canonical form.





Example 2 (Pascalian Axiom) Let A, B and C be three points on one line, and P, Q, and R be three points on another line. If $AQ \parallel RB$ and $BP \parallel QC$ then $AP \parallel RC$.

Point order: A, B, P, Q, C, R.

Hypotheses: coll(A, B, C), coll(P, Q, R), para(A, Q, B, R), para(B, P, C, Q).
Conclusion: para(C, R, A, P).

The Machine Proof
$\frac{S_{CPA}}{S_{RPA}}$
($S_{CPA} = S_{CBP} + S_{BAP} = S_{QPB} - S_{PBA}$, because collinear($A$,$B$,$C$), $CQ \parallel BA$. (L10))
($S_{RPA} = S_{RQA} + S_{QPA}S_{QPA} - S_{QBA}$, because collinear($P$,$Q$,$R$), $RB \parallel QP$. (L10))
$= \frac{S_{QPB}-S_{PBA}}{S_{QPA}-S_{QBA}}$
($S_{QPB} - S_{PBA} = S_{QPAB}$. (L14))
($S_{QPA} - S_{QBA} = S_{QPAB}$. (L14))
$= \frac{S_{QPAB}}{S_{QPAB}}$
(Simplification: removing the factor: S_{QPAB}.)
$= 1$

Example 3 (Generalization of the Butterfly Theorem) The cross ratio of four points on a circles with respect to any points on the circle is constant.

Point order: $A, B, C, D, E, E_1, F, G, F_1, G_1$.
Hypotheses: cyclic(A, B, C, D, E, E_1), coll(A, B, F, G, F_1, G_1), coll(F, E, D), coll(G, E, C), coll(F_1, E_1, D), coll(G_1, E_1, C).
Conclusion: $\frac{\overline{AF}}{\overline{BF}} \cdot \frac{\overline{BG}}{\overline{AG}} = \frac{\overline{AF_1}}{\overline{BF_1}} \cdot \frac{\overline{BG_1}}{\overline{AG_1}}$.

The Machine Proof
$\frac{\overline{F_1B}}{\overline{F_1A}} \cdot \frac{\overline{GB}}{\overline{GA}} / \frac{\overline{G_1B}}{\overline{G_1A}} \cdot \frac{\overline{FB}}{\overline{FA}}$
($\frac{\overline{F_1B}}{\overline{F_1A}} = \frac{S_{E_1DB}}{S_{E_1DA}}$, because collinear($A$,$B$,$F_1$), collinear($D$,$E_1$,$F_1$). (L6))
($\frac{\overline{G_1B}}{\overline{G_1A}} = \frac{S_{E_1CB}}{S_{E_1CA}}$, because collinear($A$,$B$,$G_1$), collinear($C$,$E_1$,$G_1$). (L6))
$= \frac{\frac{\overline{GB}}{\overline{GA}} \cdot S_{E_1DB} \cdot S_{E_1CA}}{\frac{\overline{FB}}{\overline{FA}} \cdot S_{E_1DA} \cdot S_{E_1CB}}$
($\frac{\overline{GB}}{\overline{GA}} = \frac{S_{ECB}}{S_{ECA}}$, because collinear($A$,$B$,$G$), collinear($C$,$E$,$G$). (L6))
($\frac{\overline{FB}}{\overline{FA}} = \frac{S_{EDB}}{S_{EDA}}$, because collinear($A$,$B$,$F$), collinear($D$,$E$,$F$). (L6))
$= \frac{S_{E_1DB} \cdot S_{E_1CA} \cdot S_{EDA} \cdot S_{ECB}}{S_{E_1DA} \cdot S_{E_1CB} \cdot S_{EDB} \cdot S_{ECA}}$
($S_{E_1DB} = \frac{\overline{E_1D} \cdot \overline{E_1B} \cdot \overline{DB}}{2\delta}$, because cyclic($E_1$,$D$,$B$). (L11))
($S_{E_1DA} = \frac{\overline{E_1D} \cdot \overline{E_1A} \cdot \overline{DA}}{2\delta}$, because cyclic($E_1$,$D$,$A$). (L11))
$= \frac{\overline{E_1D} \cdot \overline{E_1B} \cdot S_{E_1CA} \cdot S_{EDA} \cdot S_{ECB} \cdot \overline{DB} \cdot \delta}{\overline{E_1D} \cdot \overline{E_1A} \cdot S_{E_1CB} \cdot S_{EDB} \cdot S_{ECA} \cdot \overline{DA} \cdot \delta}$
(Simplification: removing the factors: $\overline{E_1D}, \delta$.)

$= \frac{\overline{E_1B} \cdot S_{E_1CA} \cdot S_{EDA} \cdot S_{ECB} \cdot \overline{DB}}{\overline{E_1A} \cdot S_{E_1CB} \cdot S_{EDB} \cdot S_{ECA} \cdot \overline{DA}}$

($S_{E_1CA} = \frac{\overline{E_1C} \cdot \overline{E_1A} \cdot \overline{CA}}{2\delta}$, because cyclic($E_1$,$C$,$A$). (L11))
($S_{E_1CB} = \frac{\overline{E_1C} \cdot \overline{E_1B} \cdot \overline{CB}}{2\delta}$, because cyclic($E_1$,$C$,$B$). (L11))

$= \frac{\overline{E_1C} \cdot \overline{E_1B} \cdot \overline{E_1A} \cdot S_{EDA} \cdot S_{ECB} \cdot \overline{DB} \cdot \overline{CA} \cdot \delta}{\overline{E_1C} \cdot \overline{E_1B} \cdot \overline{E_1A} \cdot S_{EDB} \cdot S_{ECA} \cdot \overline{DA} \cdot \overline{CB} \cdot \delta}$

(Simplification: removing the factors: $\overline{E_1C}, \overline{E_1B}, \overline{E_1A}, \delta$.)

$= \frac{S_{EDA} \cdot S_{ECB} \cdot \overline{DB} \cdot \overline{CA}}{S_{EDB} \cdot S_{ECA} \cdot \overline{DA} \cdot \overline{CB}}$

($S_{EDA} = \frac{\overline{ED} \cdot \overline{EA} \cdot \overline{DA}}{2\delta}$, because cyclic($E$,$D$,$A$). (L11))
($S_{EDB} = \frac{\overline{ED} \cdot \overline{EB} \cdot \overline{DB}}{2\delta}$, because cyclic($E$,$D$,$B$). (L11))

$= \frac{\overline{ED} \cdot \overline{EA} \cdot S_{ECB} \cdot \overline{DB} \cdot \overline{DA} \cdot \overline{CA} \cdot \delta}{\overline{ED} \cdot \overline{EB} \cdot S_{ECA} \cdot \overline{DB} \cdot \overline{DA} \cdot \overline{CB} \cdot \delta}$

(Simplification: removing the factors: $\overline{ED}, \overline{DB}, \overline{DA}, \delta$.)

$= \frac{\overline{EA} \cdot S_{ECB} \cdot \overline{CA}}{\overline{EB} \cdot S_{ECA} \cdot \overline{CB}}$

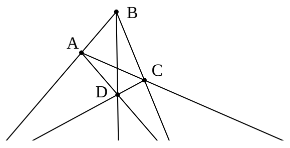
($S_{ECB} = \frac{\overline{EC} \cdot \overline{EB} \cdot \overline{CB}}{2\delta}$, because cyclic($E$,$C$,$B$). (L11))
($S_{ECA} = \frac{\overline{EC} \cdot \overline{EA} \cdot \overline{CA}}{2\delta}$, because cyclic($E$,$C$,$A$). (L11))

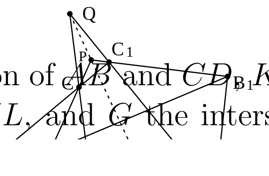
$= \frac{\overline{EC} \cdot \overline{EB} \cdot \overline{EA} \cdot \overline{CB} \cdot \overline{CA} \cdot \delta}{\overline{EC} \cdot \overline{EB} \cdot \overline{EA} \cdot \overline{CB} \cdot \overline{CA} \cdot \delta}$

(Simplification: removing the factors: $\overline{EC}, \overline{EB}, \overline{EA}, \overline{CB}, \overline{CA}, \delta$.)

$= 1$

Ndg conditions: D,E_1,A are not collinear; C,E_1,A are not collinear; C,E,A are not collinear; D,E,A are not collinear.





Example 4 (Harmonic Set) Let L be the intersection of AB and CD, K the intersection of AD and BC, F the intersection of BD and KL, and G the intersection of AC and KL. Then $\frac{LF}{KF} = \frac{LG}{GK}$.

The Machine Proof
$-\frac{\overline{FK}}{\overline{FL}} / \frac{\overline{GK}}{\overline{GL}}$

($\frac{\overline{FK}}{\overline{FL}} = -\frac{\overline{KA}}{\overline{DA}} / \frac{\overline{LC}}{\overline{DC}}$, because FCA is the cevain triangle of point B for KLD (the Ceva Theorem).)
($\frac{\overline{GK}}{\overline{GL}} = \frac{\overline{KA}}{\overline{DA}} / \frac{\overline{LC}}{\overline{DC}}$, because GCA is a transversal for triangle KLD (the Menelaus Theorem).)

$= \frac{\overline{KA}}{\overline{DA}} \cdot \frac{\overline{LC}}{\overline{DC}} / \frac{\overline{KA}}{\overline{DA}} \cdot \frac{\overline{LC}}{\overline{DC}}$

(Simplification: removing the factors: $\frac{\overline{KA}}{\overline{DA}}, \frac{\overline{LC}}{\overline{DC}}$.)

$= 1$

Ndg conditions: B,K,L are not collinear; K,G,C are not collinear; D,G,C are not collinear.

Example 5 (Desargues' Theorem) Given two triangles ABC, $A_1B_1C_1$, if the three lines AA_1, BB_1, CC_1 meet in a point, S, the three points $P = BC \cap B_1C_1$, $Q = CA \cap C_1A_1$, $R = AB \cap A_1B_1$ lie on a line.

Point order: $O, A, B, C, A_1, B_1, C_1, P, Q, S, R$.

Hypotheses: coll(A_1, A, O), coll(B_1, B, O), coll(C_1, C, O), coll(B, C, P), coll(B_1, C_1, P), coll(A, C, Q), coll(A_1, C_1, Q), coll(A, B, S), coll(A_1, B_1, S), coll(P, Q, R), coll(A, B, R).

Conclusion: $\frac{\overline{AS}}{\overline{BS}} = \frac{\overline{AR}}{\overline{BR}}$.

The Machine Proof

$\frac{\overline{RB}}{\overline{RA}} / \frac{\overline{SB}}{\overline{SA}}$

($\frac{\overline{RB}}{\overline{RA}} = \frac{\overline{QC}}{\overline{QA}} / \frac{\overline{PC}}{\overline{PB}}$, because RQP is a transversal for triangle BAC (the Menelaus Theorem).)

($\frac{\overline{SB}}{\overline{SA}} = \frac{\overline{B_1B}}{\overline{B_1O}} / \frac{\overline{A_1A}}{\overline{A_1O}}$, because SA_1B_1 is a transversal for triangle BAO (the Menelaus Theorem).)

$= \frac{\overline{QC}}{\overline{QA}} \cdot \frac{\overline{A_1A}}{\overline{A_1O}} / \frac{\overline{PC}}{\overline{PB}} \cdot \frac{\overline{B_1B}}{\overline{B_1O}}$

($\frac{\overline{QC}}{\overline{QA}} = \frac{\overline{C_1C}}{\overline{C_1O}} / \frac{\overline{A_1A}}{\overline{A_1O}}$, because QA_1C_1 is a transversal for triangle CAO (the Menelaus Theorem).)

($\frac{\overline{PC}}{\overline{PB}} = \frac{\overline{C_1C}}{\overline{C_1O}} / \frac{\overline{B_1B}}{\overline{B_1O}}$, because PB_1C_1 is a transversal for triangle CBO (the Menelaus Theorem).)

$= \frac{\overline{C_1C}}{\overline{C_1O}} \cdot \frac{\overline{B_1B}}{\overline{B_1O}} \cdot \frac{\overline{A_1A}}{\overline{A_1O}} / \frac{\overline{C_1C}}{\overline{C_1O}} \cdot \frac{\overline{B_1B}}{\overline{B_1O}} \cdot \frac{\overline{A_1A}}{\overline{A_1O}}$

(Simplification: removing the factors: $\frac{\overline{C_1C}}{\overline{C_1O}}, \frac{\overline{B_1B}}{\overline{B_1O}}, \frac{\overline{A_1A}}{\overline{A_1O}}$.)

$= 1$

Ndg conditions: B,R,Q are not collinear; C,R,Q are not collinear; B,S,A_1 are not collinear; O,S,A_1 are not collinear; C,Q,A_1 are not collinear; O,Q,A_1 are not collinear; C,P,B_1 are not collinear; O,P,B_1 are not collinear.