A GEOMETRY THEOREM PROVER FOR MACINTOSHES

Shang-Ching Chou

Department of Computer Sciences
University of Texas at Austin
Austin, Texas 78712-1188

TR-91-08 March 1991

A Geometry Theorem Prover for Macintoshes*

Shang-Ching Chou

Department of Computer Sciences
The University of Texas at Austin, Austin, Texas 78712 USA

Abstract. This report primarily serves as a manual for a geometry theorem prover implemented for the Macintosh. The prover is based on the algebraic method introduced by Wen-Tsün Wu and on a prover developed on Symbolics Lisp Machines by the author. It addresses a subset of geometry statements which involve equalities only and can prove a subset of theorems that the original prover proved. Within its scope, it is powerful enough to prove many theorems hard for human to prove such as Pappus' theorem, Simson's theorem, Pascal's theorem, the nine-point theorem, Feuerbach's theorem, etc. There is also a discussion of the role of non-degenerate conditions in a geometry statement.

Keywords. Geometry theorem proving, mechanical method, algebraic method, Wu's method, non-degenerate condition, constructive geometry statement, Euclidean geometry, metric geometry.

* The work reported here was supported in part by the NSF Grant CCR-9002362.

1. Introduction

This experimental geometry prover for Macintoshes was originally written as a demonstration for the special session in automated theorem proving at the AMS annual meeting, San Francisco, 1991. The prover is based on a prover developed on Symbolics Lisp Machines and can prove a subset of theorems that the original prover proved. It is powerful enough to prove many hard theorems such as Pappus' theorem, Simson's theorem, Pascal's theorem, the nine-point theorem, Feuerbach's theorem, Steiner's theorem, etc. With a further extension, it is expected to prove over 90% of the theorems that the original prover proved, including Morley's trisector theorem.

The prover is based on the algebraic method introduced by Wen-Tsün Wu ([5] and [6]) and further developed by the author. For an introductory exposition, see [2]. For a complete exposition, see [3].

However, this prover is more like the one described in [4]. It is mainly for a class of geometry statements of constructive type described in [4] (also see the Appendix). At present, it has proved about 400 theorems.

The Mac plus or SE version can be obtained by writing to

Shang-Ching Chou
Department of Computer Sciences
University of Texas at Austin
Austin, Texas 78712

with a self-addressed envelope, a sufficient postage on it, and an 800K disk (or $1.50 for the cost of the disk). The request after July 1991 should be sent to

Shang-Ching Chou
Department of Computer Science
Wichita State University
Wichita, Kansas 67208.

This documentation can be obtained by writing to

Technical Report Center
Department of Computer Sciences (Tay 2.124)
University of Texas at Austin
Austin, Texas 78712-1188.

An additional cost may be charged from the technical report center.

In the disk, there are a system fold, the program, a file (named "sample.lisp") containing 12 sample examples, and this documentation in the TEX form. The disk can run on the mac plus and SE(SE/30) without using a hard disk.

This preliminary documentation is mainly to explain how to use the prover (Section 2–5). In the Appendix, we shall explain how to produce the construction sequence from the input and how to generate non-degenerate conditions in geometric form mechanically.

2. Input (or Specification of a Geometry Statement)

The first step to convert a geometry statement into its algebraic form is to assign coordinates to the points in the statement. On our prover this can be done either manually or automatically. Here we show how to specify the input (the geometry statement) for the program to assign coordinates automatically. *The statement must be specified in a constructive way.*

Most theorems (involving only equalities) in elementary geometry can be stated in a constructive way, i.e., given a set of arbitrarily chosen points, new points are added in a constructive way by introducing one or two geometry conditions (constructions) for each point, such as taking an arbitrary point on a line or on a circle (one condition), or taking intersections of two lines or one line and one circle, or two circles (two conditions). The conclusion is a (equality) relation among these points.

The user *must* have a clear idea about the order in which the points are introduced.

The input is in the Lisp list form (though the present program is written in Pascal and has nothing to do with Lisp; we use this input format for consistency with the input of our original prover in Lisp).

The first element in the input is a list beginning with the keyword "point-order" or "cons-seq", followed by the points in the constructive order (arbitrarily chosen points are first). The following elements (but the last) of the input are the geometry conditions to construct the points. They are the equality part of the hypotheses of the geometry statement. The order of these geometry conditions is not important. They can be arranged in any order. But it is recommended that they be written according to the point order, which is important. The last element is the geometry condition representing the conclusion to be proved.

Following are the four examples in first Chapter of [3] (the actually coordinate selections might be slightly different from those in [3]).

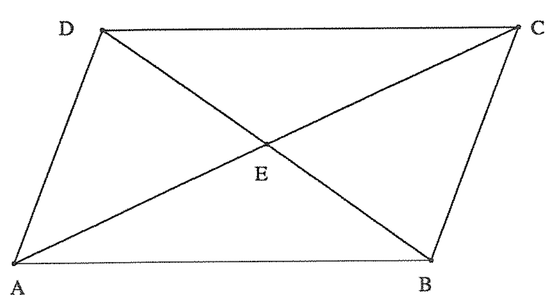

Figure 1

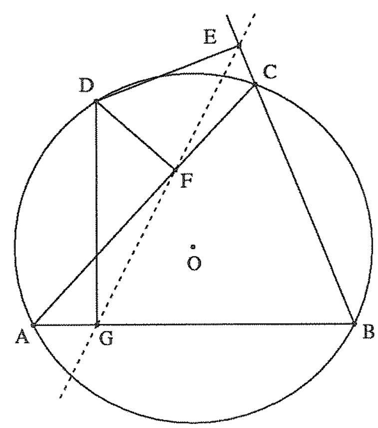

Figure 2

Examples 1. Let $ABCD$ be a parallelogram (i.e, $AB \parallel CD$, $BC \parallel AD$), E be the intersection of the diagonals AC and BD. Show $AE \equiv CE$ (Figure 1).

The input is

```
(setq paral
   '((cons-seq A B C D E)
     (parallel A B C D)
     (parallel B C A D)
     (collinear A E C)
     (collinear B E D)
     (eqdistance A E C E)))
```

Remark. The letter case is not important. Also "setq" and "defvar" are interchangeable.

Example 2. (Simson's theorem) Let D be a point on the circumscribed circle (O) of triangle ABC. From D three perpendiculars are drawn to the three sides BC, CA and AB of $\triangle ABC$. Let E, F and G be the three feet respectively. Show that E, F and G are collinear (Figure 2).

The input is

```
(setq simson
   '((cons-seq A B C O D E F G)
     (collinear A B G)
     (collinear B C E)
     (collinear A C F)
     (perpendicular E D B C)
     (perpendicular F D A C)
     (perpendicular G D A B)
     (eqdistance A O B O)
     (eqdistance A O C O)
     (eqdistance A O D O)
     (collinear E F G)))
```

Here we deliberately write the hypothesis conditions not in the order in which the points are introduced. Permuting those conditions will not affect the constructions (hence the selection of coordinates in any way). Try it if you wish. But if you change the point order, then the constructions will be changed. E.g., if you put point O first, then the program can figure out you first have a circle, then have three points on the circle, etc. For the above input, the constructions are (You can select the Item "Show Thm in Constructions" in the Info Menu to show them on the screen):

```
Points A, B, C, are arbitrarily chosen
O is on B(A B) and B(A C)
D is on R(O, A O)
E is on L(B C) and T(D, B C)
F is on L(A C) and T(D, A C)
G is on L(A B) and T(D, A B).
```

The most common lines which can be constructed from the preceding points are one of the following four types:

L(A B): the line passing through points A and B.

P(A, B C): the line passing through A and parallel to line BC.

T(A, B C): the line passing through A and perpendicular to line BC.

B(A B): the perpendicular bisector of the segment AB.

The most common circles which can be constructed from the preceding points are one of the following two types:

R(O, A B): the circle with the center O and radius AB.

C(A B C): the circle passing through the three points A, B, and C.

If you change the first line of input to

(cons-seq $O\ A\ B\ C\ D\ E\ F\ G$)

then the construction sequence is

Points O, A, are arbitrarily chosen
B is on R(O, A O)
C is on R(O, A O)
D is on R(O, A O)
E is on L(B C) and T(D, B C)
F is on L(A C) and T(D, A C)
G is on L(A B) and T(D, A B).

Also select the Item "Show Degenerate Conds" in the Info Menu to see the differences between these two inputs (Also see Section 4).

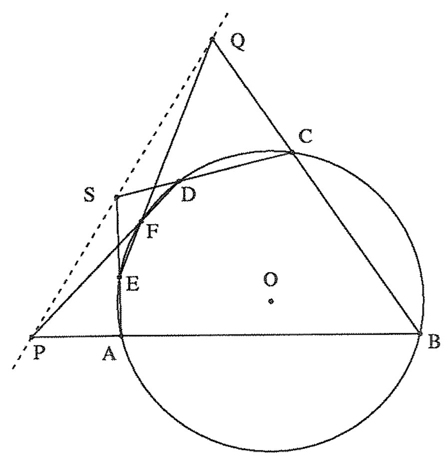

Figure 3

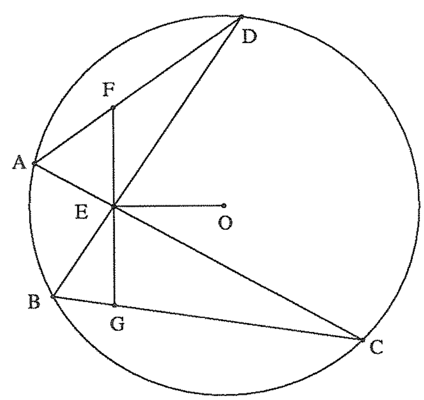

Figure 4

Example 3. (Pascal's Theorem). Let A, B, C, D, F and E be six points on a circle (O). Let $P = AB \cap DF$, $Q = BC \cap FE$ and $S = CD \cap EA$. Show that P, Q and S are collinear (Figure 3).

The input is

```
(setq pascal
  '((cons-seq A O B C D F E P Q S)
    (eqdistance A O B O)
    (eqdistance A O C O)
```

```
(eqdistance A O D O)
(eqdistance A O E O)
(eqdistance A O F O)
(collinear A B P)
(collinear D F P)
(collinear B C Q)
(collinear F E Q)
(collinear C D S)
(collinear E A S)
(collinear P Q S)))
```

Example 4. (The Butterfly Theorem). A, B, C and D are four points on circle (O). E is the intersection of AC and BD. Through E draw a line perpendicular to OE, meeting AD at F and BC at G. Show that $FE \equiv GE$ (Figure 4).

The input is

```
(setq butterfly
  '((cons-seq E O A B C D F G)
   (eqdistance O A O B)
   (eqdistance O A O C)
   (eqdistance O A O D)
   (perpendicular O E E F)
   (collinear A E C)
   (collinear B E D)
   (collinear A D F)
   (collinear F E G)
   (collinear B C G)
   (midpoint F E G)))
```

In the file "sample.lisp", there are another 8 theorems. These 12 theorems are enough to test a new geometry theorem prover based on similar algebraic methods.

3. A list of Geometry Conditions for the Input

Let let $A = (x_1, y_1)$, $B = (x_2, y_2)$, $C = (x_3, y_3)$ and $D = (x_4, y_4)$.

1. (collinear A B C) (abbr. L, coll)

2. (parallel A B C D) (abbr. P, paral)

AB is parallel to CD.

3. (perpendicular A B C D) (abbr. T, perp)

AB is perpendicular to CD.

4. (eqdistance A B C D) (abbr. E, =)

$AB \equiv CD$.

5. (perp-bisect A B C) (abbr. I)

A is on the perpendicular bisector of BC, or equivalently $AB \equiv AC$.

6. (midpoint $A\ B\ C$) (abbr. M)

B is the midpoint of AC.

7. (o-ratio $A\ B\ C\ n\ m$) (abbr. O)

A, B, and C and collinear and the ratio of the oriented segments AB and BC are n/m. This condition (or midpoint) usually generates two polynomials. For example, (o-ratio $A\ B\ C\ -1$ 2) generates:

$(x_1 - x_2) = (-1/2)(x_2 - x_3)$ and $(y_1 - y_2) = (-1/2)(y_2 - y_3)$.

8. (eqangle $A\ B\ C\ E\ F\ G$) (abbr. A).

$\angle ABC \equiv \angle EFG$. The definition of angle congruence in unordered geometry is very subtle, see [3] for details.

9. (cocircle $A\ B\ C\ D$) (abbr. C)

A, B, C, and D are on the same circle.

10. (l–c–tangent $A\ B\ C\ D\ O$) (abbr. L–C)

Line AB is tangent to the circle $(O, (CD))$ (i.e., with the center O and radius CD).

11. (c–c–tangent $A\ B\ O\ C\ D\ O_1$) (abbr. C–C)

Two circles R(O, $A\ B$) and R(O_1, $C\ D$) are tangent.

12. (eq-product $A\ B\ CD\ E\ F\ G\ H$) (abbr. Z)

$AB \cdot CD = EF \cdot GH$.

There are other less commonly used conditions. With further expansions, we can cover most of 512 geometry statements proved in [3]. Based on our experience, we believe that we can prove over 90% of those theorems with 1M byte memory.

4. On Non-Degenerate Conditions

The above geometric conditions (and their algebraic equivalents) usually include degenerate cases. For example, the exact meaning of (parallel $A\ B\ C\ D$) is $[(A = B) \vee (C = D) \vee (AB \parallel CD)]$. It is not instructive to exclude the cases when $A = B$ and $C = D$. By doing so, we still cannot exclude all non-degenerate conditions necessary for a geometry statement to be valid.

E.g., for Example 1, the statement

$$\begin{aligned} \forall A \cdots E\ [[&(\text{parallel } A\ B\ C\ D) \wedge (\text{parallel } B\ C\ A\ D) \\ &\wedge (\text{collinear } A\ E\ C) \wedge (\text{collinear } B\ E\ D)] \\ &\Rightarrow (\text{eqdistance } A\ E\ C\ E)] \end{aligned}$$

is not a valid statement. We need other non-degenerate conditions.

In the current research, there are two Approaches or Formulations to prove geometry statements.

Formulation F1 is to prove the above statement to be generally or generically true, at the same time giving sufficiently many non-degenerate conditions. Those conditions usually are in algebraic form. However, for a class of geometry statements of constructive type specified in [1] and [4], we can generate sufficiently many non-degenerate conditions in *geometric form*. In [1] and [4], we have proved a theorem stating that if a statement is still not valid under these machine generated non-degenerate conditions, then the statement cannot be valid no matter how many more reasonable non-degenerate conditions are added.

After selecting a theorem from the Menu "Theorems", click the Item "Show degenerate Conds" in the Info Menu, you will get those degenerate conditions.

In Formulation F2, users have to specify all non-degenerate conditions manually, and then the prover only needs to answer "true" or "false". Finding all non-degenerate conditions is very subtle, especially because the methods we use (Wu's method or the Gröbner basis method) are complete only for algebraically closed fields. We haven't built methods for Formulation F2 on this Mac version yet.

Try Simson's theorem in the file "sample.lisp". If you don't understand isotropic lines, it is fine, because such lines do not exist in Euclidean geometry. But they do exist in general unordered geometries. Try the Butterfly theorem, and you will find (by clicking the item "Show degenerate conds" in the Info Menu) an unexpected degenerate condition "OE is perpendicular to AD", which is necessary even for Euclidean geometry.

5. How to Use the Prover

First open the prover icon. Then click the File Menu and choose the item "Open Thm File". Then select the file "sample.lisp". Then there will appear a new Menu "theorems" containing the theorem names in that file. Check any theorem name in that menu, then the program reads the input of that theorem, figures out the constructions and degenerate conditions, and assigns coordinates to the points. You can check the corresponding items in the Info Menu.

Here is a brief account of the items in all menus.

1. File Menu.

"Prove Thm": Prove the theorem which has been checked in the Theorems Menu.

"Draw Thm": Draw the figure of the theorem which has been checked in the Theorems Menu.

"Prove All": Prove all theorems in the Theorems Menu.

"Open Thm file": Open a file containing theorems. The file name must be ended with ".lisp".

"Demo Prove": Show the detailed steps of the proof of the theorem checked. You must try it. It is a very nice feature.

2. Edit Menu.

Selecting these items does not cause any action. They are reserved for later implementation.

3. Drawings Menu.

"Join Line": Join a line by selecting two points. To select a point, click the mouse on the point. When a point is near enough and is selected, a double beep will sound.

"Draw R–Circle": Draw a circle by selecting its center and one point on the circle.

"Draw C-circle": Draw a circle by selecting three points on the circle.

"Move Figure": Move the figure by clicking the mouse. (The new origin is at the mouse position).

"Save Figure": Save the drawn figure to a file ending with ".fig".

"Draw Saved Figure": draw the figure saved in the file ending with ".fig" if there is one.

4. Options Menu.

Most items involve technical details of the proof method. See e.g., page 89 of [3].

5. Info Menu. Show various information pertaining to a checked theorem.

If you want to prepare your own theorems to prove, use a text editor, and write the theorems in the form shown in Examples 1–4 in a file such as "thms.lisp". Then the corresponding figure information will be placed in the file "thms.fig". If you want to delete a saved figure, you have to open that file with a text editor and delete that figure information (it is self-evident, once you look at that file).

If you really want to prove a hard theorem, check "Simplify hyp-polys" and uncheck "Successive Division" in the Options Menu. As a convention, the first point in "cons-seq" is the origin of the coordinate system, and the second point is on the x–axis. Selection of the origin and the x–axis affects the sizes of polynomials produced in a proof.

Acknowledgment. The author wishes to thank R. S. Boyer and J S. Moore for their advice and encouragement to write a Mac version of my geometry theorem prover and thank N. McPhee for his help and suggestions in writing this Mac program. For over 8 years, Boyer and Moore have been constantly encouraging the author in studying geometry theorem proving. Without such encouragements, the pace of recent development in this area would have been much slower.

References

[1] S.C. Chou, "Proving and Discovering Theorems in Elementary Geometries Using Wu's Method", PhD Thesis, Department of Mathematics, University of Texas, Austin (1985).

[2] S.C. Chou, "An Introduction to Wu's Method for Mechanical Theorem Proving in Geometry", *Journal of Automated Reasoning,* 4 (1988), 237–267.

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

[4] S.C. Chou and X.S. Gao, "A Class of Geometry Statements of Constructive Type and Geometry Theorem Proving", TR-89-37, Computer Sciences Department, The University of Texas at Austin, November, 1989.

[5] Wu Wen-tsün, "On the Decision Problem and the Mechanization of Theorem Proving in Elementary Geometry", *Scientia Sinica* **21** (1978), 157-179.

[6] Wu Wen-tsün, "Basic Principles of Mechanical Theorem Proving in Geometries", *J. of Sys. Sci. and Math. Sci.* **4(3)**, 1984, 207-235, republished in *Journal of Automated Reasoning* **2(4)** (1986), 221-252.

6. Appendix

In this Appendix, we shall explain how to produce the construction sequence for a geometry statement from the input and how to generate non-degenerate conditions in geometric form. We define a class of geometric statements of constructive type, called Class C, as follows.

6.1. Definition of Class C

Most theorems in elementary geometry can be described in a constructive way: given a certain arbitrary points, lines, circles and points on these circles and lines, new points, lines and circles are constructed step by step using geometric constructions such as taking the intersection of two lines, an intersection of a line and a circle, or an intersection of two circles. In this subsection, we use the natural language to give a definition of such a statement. In Section 6.3, we will give the precise formula of such a statement using geometric predicates.

First, let us give "circle" a definition. A *circle* h is a pair of a point O and a segment (AB): $h = (O,(AB))$. Two circles $(O,(AB))$ and $(P,(CD))$ are equal if $O = P$ and congruent(A, B, C, D). O is called the center of the circle and (AB) the radius. A point P is on circle $(O,(AB))$ if congruent(O, P, A, B) (for the precise meaning of the predicate "congruent", see 6.2).

Let Π be a finite set of points. We say line l is constructed *directly* from Π if

(i) The line l joins two points A and B in Π. We denote it by $l = L(AB)$; or

(ii) The line l passes through one point C in Π and parallel to a line joining two points A and B in Π. We denote it by $l = P(C, AB)$; or

(iii) The line l passes through one point C in Π and perpendicular to a line joining two points

A and B in Π. We denote it $l = T(C, AB)$; or

(iv) The line l is the perpendicular–bisector of AB with A and B in Π. We denote it by $l = B(AB)$.

A line l constructed directly from Π is *well defined* if the two points A and B mentioned above are distinct.

Likewise, we say a circle $c = (O, (AB))$ is constructed directly from Π if points O, A and B are in Π. The lines and circles constructed directly from Π are said to be *in* Π, for brevity.

Definition. A geometry statement is of constructive type or in Class C if the points, lines, and circles in the statement can be constructed in a definite prescribed manner using the following ten constructions, assuming Π to be the set of points already constructed so far:

Construction 1. Taking an arbitrary point.

Construction 2. Drawing an arbitrary line. This can be reduced to taking two arbitrary points.

Construction 3. Drawing an arbitrary circle. This can be also reduced to taking two or three arbitrary points.

Construction 4. Drawing an arbitrary line passing through a point in Π. This can be reduced to taking an arbitrary point.

Construction 5. Drawing an arbitrary circle knowing its center in Π. This can be also reduced to taking one or two arbitrary points.

Construction 6. Taking an arbitrary point on a line in Π.

Construction 7. Taking an arbitrary point on a circle in Π.

Construction 8. Taking the intersection of two lines in Π.

Construction 9. Taking an intersection of a line and a circle in Π.

Construction 10. Taking an intersection of two circles in Π.

The conclusion is a certain (equality) relation among the points thus constructed.

In the actual prover [1] and this simple version for Macintoshes, we have included more constructions such as taking midpoints and constructions involving angle congruence, the radical axis of two circles, taking a point on a circle knowing three points on the circle, etc.

Example (6.1). Simson's theorem can be specified as a statement in Class C by the following *construction sequence*:

Points A, B, and C are arbitrarily chosen;	construction 1
$O = B(AB) \cap B(AC)$;	construction 8
D is on circle $(O, (OA))$;	construction 7
$E = T(D, BC) \cap L(BC)$;	construction 8
$F = T(D, AC) \cap L(AC)$;	construction 8
$G = T(D, AB) \cap L(AB)$.	construction 8

The Butterfly theorem can be specified as a statement in Class C by the following construction sequence:

O and A are arbitrarily chosen;	construction 1
B is on $(O,(OA))$;	construction 7
C is on $(O,(OA))$;	construction 7
D is on $(O,(OA))$;	construction 7
$E = L(AC) \cap L(BD)$;	construction 8
$F = L(AD) \cap T(E, OE)$;	construction 8
$G = L(EF) \cap L(BC)$.	construction 8

In the above examples, we use a *construction sequence* to express a statement in Class C. The construction sequence still does not specify what exact non-degenerate conditions are needed for a geometric statement in Class C. We will soon present an algorithm for generating non-degenerate conditions for a statement in Class C knowing its construction sequence. Before presenting the algorithm, we first specify what exact geometric predicates we use.

6.2. The Basic Predicates

In order to describe the logical formula of a statement in Class C, we only need four basic (non-logical) predicates: "collinear(A,B,C)", "parallel(A,B,C,D)", "perpendicular(A,B,C,D)", "congruent(A,B,C,D)".[1] The first thing we should emphasize is that these predicates do include degenerate cases. To be more precise, let $A = (x_1, y_1)$ $B = (x_2, y_2)$, $C = (x_3, y_3)$ and $D = (x_4, y_4)$.

(1) Predicate "collinear(A,B,C)" means that points A, B and C are on the same line; they are not necessarily distinct. Its corresponding algebraic equation is

$$(x_1 - x_2)(y_2 - y_3) - (x_2 - x_3)(y_1 - y_2) = 0.$$

(2) Predicate "parallel(A,B,C,D)" means that

$[(A = B) \vee (C = D) \vee (A, B, C, D \text{ are on the same line}) \vee (AB \parallel CD)]$. Its algebraic equation is

$$(x_1 - x_2)(y_3 - y_4) - (x_3 - x_4)(y_1 - y_2) = 0.$$

(3) Predicate "perpendicular(A,B,C,D)" means that $[(A = B) \vee (C = D) \vee (AB \perp CD)]$. Its algebraic equation is

$$(x_1 - x_2)(x_3 - x_4) + (y_1 - y_2)(y_3 - y_4) = 0.$$

(4) Predicate "congruent(A,B,C,D)" includes the cases when $A = B$ and $C = D$. Its algebraic equation is

$$(x_1 - x_2)^2 + (y_1 - y_2)^2 - (x_3 - x_4)^2 - (y_3 - y_4)^2 = 0.$$

There are many advantages of using the above predicates. Each of the above predicates corresponds to only one equation, thus its negation corresponds to only one inequation. E.g.,

[1] In our actual prover [1], [3], there are many other predicates, such as the midpoint, angle congruence, the radical axis of two circles, etc. For the complete list of all those predicates and their algebraic equations see pp.97–99 of [3].

¬parallel(A,B,C,D) is "$(A \neq B) \wedge (C \neq D) \wedge (A,B,C,D$ are not on the same line$) \wedge \neg(AB \parallel CD)$". Its corresponding inequation is

$$(x_1 - x_2)(y_3 - y_4) - (x_3 - x_4)(y_1 - y_2) \neq 0,$$

which is *the exact non-degenerate condition we want for intersecting two lines AB and CD: they have only one common point.* Note that this condition implies the condition $(A \neq B \wedge C \neq D)$. We can use the negations of the four predicate in a convenient way. E.g., ¬perpendicular(A,B,A,B) means $A \neq B$ and AB is non-isotropic, i.e., ¬isotropic(A,B), or $(x_1 - x_2)^2 + (y_1 - y_2)^2 \neq 0$. Here we define a new predicate "isotropic(A,B)" to be perpendicular(A,B,A,B).

Now we are in a position to present our algorithm for generating non-degenerate conditions for statements in Class C.

6.3 Mechanical Generation of Non-Degenerate Conditions for Class C

For a statement in Class C, we can generate non-degenerate conditions following the construction sequence step by step. Suppose we have already generated a set of non-degenerate conditions DS under the previous constructions. Let HS be the set of the equation hypotheses under the previous constructions, and Π be the set of points constructed so far. The next construction is one of the ten constructions in Section 6.1. First we add the point(s) to be constructed to the set Π. Since the first five constructions are reduced to taking arbitrary points, nothing is added to HS or DS. Thus we assume the next construction is one of constructions 6–10. We use abbreviations coll(), perp(), para() and cong() for predicates collinear(), perpendicular(), parallel() and congruent(), respectively.

Construction 6. Taking an arbitrary point D on a line l in Π. There are four kinds of lines in Π.

(i) $l = L(AB)$.

$HS := \{\text{coll}(A,B,D)\} \cup HS$; $DS := \{A \neq B\} \cup DS$.

(ii) $l = P(C,AB)$.

$HS := \{\text{para}(A,B,C,D)\} \cup HS$; $DS := \{A \neq B\} \cup DS$.

(iii) $l = T(C,AB)$.

$HS := \{\text{perp}(A,B,C,D)\} \cup HS$; $DS := \{A \neq B\} \cup DS$.

(iv) $l = B(AB)$.

$HS := \{\text{cong}(A,D,B,D)\} \cup HS$; $DS := \{A \neq B\} \cup DS$.

Construction 7. Taking an arbitrary point A on a circle $(B,(CD))$ in Π.

$HS := \{\text{cong}(A,B,C,D)\} \cup HS$.

Construction 8. Taking the intersection I of two lines in Π.

Since there are four types of lines in Π, there are 10 types of intersections: types LL, LP, LT, LB, PP, PT, PB, TT, TB, and BB.

Let the two lines be given by the following equations:

$$l_1 : a_1x + b_1y + c_1 = 0,$$
$$l_2 : a_2x + b_2y + c_2 = 0.$$

The elegance of our approach is that for all 10 types of intersections, the only non-degenerate condition in algebraic form is $\Delta = a_1b_2 - a_2b_1 \neq 0$.

Case 8.1. Type LL: $I = L(AB) \cap L(CD)$.

$HS := \{\text{coll}(A, B, I), \text{coll}(C, D, I)\} \cup HS$; $DS := \{\neg\text{para}(A, B, C, D)\} \cup DS$.

In the algebraic form, this is equivalent to $\Delta = a_1b_2 - a_2b_1 \neq 0$. Note that this condition implies $A \neq B$ and $C \neq D$.

Case 8.2. Type LP: $= L(AB) \cap P(E, CD)$.

$HS := \{\text{coll}(A, B, I), \text{para}(C, D, E, I)\} \cup HS$; $DS := \{\neg\text{para}(A, B, C, D)\} \cup DS$. In the special case,

Case 8.2.1. If $B = D$, then instead, DS should be $DS := \{\neg\text{coll}(A, B, C)\} \cup DS$.

Case 8.3. Type LT: $I = L(AB) \cap T(E, CD)$.

$HS := \{\text{coll}(A, B, I), \text{perp}(C, D, E, I)\} \cup HS$; $DS := \{\neg\text{perp}(A, B, C, D)\} \cup DS$. (See the Butterfly theorem). In the special cases,

Case 8.3.1. If AB is parallel to CD, $\neg\text{perp}(A, B, C, D)$ is reduced to $A \neq B$, $C \neq D$, line AB is not perpendicular to AB itself. Thus instead, DS should be $DS := \{\neg\text{isotropic}(A, B)\} \cup DS$.

Case 8.3.2. Lines AB and CD are identical. $DS := \{\neg\text{isotropic}(A, B)\} \cup DS$.

Case 8.3.3. $A = C$ and $B = D$. $DS := \{\neg\text{isotropic}(A, B)\} \cup DS$. (See Condition (2.1.1) for Simson's theorem.)

Case 8.4. Type LB: $I = L(AB) \cap B(CD)$.

$HS := \{\text{coll}(A, B, I), \text{cong}(I, C, I, D)\} \cup HS$; $DS := \{\neg\text{perp}(A, B, C, D)\} \cup DS$. In the special cases,

Case 8.4.1. AB is parallel to CD. $DS := \{\neg\text{isotropic}(A, B)\} \cup DS$.

Case 8.4.2. Lines AB and CD are identical. $DS := \{A \neq B, C \neq D\} \cup DS$.

Case 8.5. Type PP: $I = P(E, AB) \cap P(F, CD)$.

$HS := \{\text{para}(A, B, E, I), \text{para}(C, D, E, I)\} \cup HS$; $DS := \{\neg\text{para}(A, B, C, D)\} \cup DS$. In the special case,

Case 8.5.1. $B = D$. $DS := \{\neg\text{coll}(A, B, C)\} \cup DS$.

Case 8.6. Type PT: $I = P(E, AB) \cap T(F, CD)$.

$HS := \{\text{para}(A, B, E, I), \text{perp}(C, D, F, I)\} \cup HS$; $DS := \{\neg\text{perp}(A, B, C, D)\} \cup DS$. In the special case,

Case 8.6.1. lines AB is parallel or identical to CD. $DS := \{\neg\text{isotropic}(A, B)\} \cup DS$.

Case 8.7. Type PB: $I = P(E, AB) \cap B(CD)$.

$HS := \{\text{para}(A, B, E, I), \text{cong}(I, C, I, D)\} \cup HS$; $DS := \{\neg\text{perp}(A, B, C, D)\} \cup DS$. In the special case,

Case 8.7.1. lines AB is parallel to CD. $DS := \{\neg\text{isotropic}(A, B)\} \cup DS$.

Case 8.8. Type TT: $I = T(E, AB) \cap T(F, CD)$.

$HS := \{\text{perp}(A, B, E, I), \text{perp}(C, D, F, I)\} \cup HS$; $DS := \{\neg\text{para}(A, B, C, D)\} \cup DS$. In the special case,

Case 8.8.1. $B = D$. $DS := \{\neg\text{coll}(A, B, C)\} \cup DS$.

Case 8.9. Type TB: $I = T(E, AB) \cap B(CD)$.

$HS := \{\text{perp}(A, B, E, I), \text{cong}(I, C, I, D)\} \cup HS$; $DS := \{\neg\text{para}(A, B, C, D)\} \cup DS$. In the special case,

Case 8.9.1. $B = C$. $DS := \{\neg\text{coll}(A, B, C)\} \cup DS$.

Case 8.10. Type BB: $I = B(AB) \cap B(CD)$.

$HS := \{\text{perp}(I, A, I, B), \text{cong}(I, C, I, D)\} \cup HS$; $DS := \{\neg\text{para}(A, B, C, D)\} \cup DS$. In the special case,

Case 8.10.1. $B = D$. $DS := \{\neg\text{coll}(A, B, C)\} \cup DS$.

Construction 9. Taking an intersection Q of a line and a circle in Π. Let the line be $L(AB)$, or $P(C, AB)$, or $T(C, AB)$, or $B(AB)$, the circle be $(O, (DE))$. $DS := \{\neg\text{isotropic}(A, B)\} \cup DS$.

If $Q = L(AB) \cap (O, (DE))$, then $HS := \{\text{coll}(A, B, Q), \text{cong}(O, Q, D, E)\} \cup HS$.

If $Q = P(C, AB) \cap (O, (DE))$, then $HS := \{\text{para}(A, B, C, Q), \text{cong}(O, Q, D, E)\} \cup HS$.

If $Q = T(C, AB) \cap (O, (DE))$, then $HS := \{\text{perp}(A, B, C, Q), \text{cong}(O, Q, D, E)\} \cup HS$.

If $Q = B(AB) \cap (O, (DE))$, then $HS := \{\text{cong}(Q, A, Q, B), \text{cong}(O, Q, D, E)\} \cup HS$.

Case 9.1. In the special case when one of the intersections, say S, of the circle and the line is already in Π. $DS := \{\neg\text{isotropic}(A, B), S \neq Q\} \cup DS$.

Construction 10. Taking an intersection Q of two circles in Π. Let the two circles be $(O, (AB))$ and $(P, (CD))$.

$HS := \{\text{cong}(O, Q, A, B), \text{cong}(P, Q, C, D)\} \cup HS$; $DS = \{\neg\text{isotropic}(O, P)\} \cup DS$. In the special case,

Case 10.1. One of the intersections is already in Π, say, S. $DS := \{\neg\text{isotropic}(O, P), S \neq Q\} \cup DS$.

Repeating the above steps until every construction is processed, finally we have two parts for the hypotheses: one is $HS = \{H_1, ..., H_r\}$, called the *equation part* of the hypotheses; the other is $DS = \{\neg D_1, ..., \neg D_s\}$, called the *inequation part* of the hypotheses and representing non-degenerate conditions of the statement. Let C be the conclusion of the statement, which is not necessarily one of the four predicates defined in Section 6.2, but whose algebraic form is a polynomial equation in the coordinates of the points in Π. Then the exact statement is[2]

$$\forall P \in \Pi(HS \wedge DS \Rightarrow C). \tag{6.2}$$

Thus according to our translation, we can denote a statement S in Class C by (HS, DS, C).

6.4. Examples

Now we use Simson's theorem and the Butterfly theorem to show how to produce the necessary non-degenerate conditions mentioned in Section 2.

Example (6.4). (Simson's Theorem and the Butterfly theorem). According to the construction sequence of Simson's theorem in Example (6.1), the non-degenerate conditions (the inequation part of the hypotheses) are

$$\left.\begin{array}{l}\neg\text{collinear}(A,B,C),\\ \neg\text{isotropic}(AB),\\ \neg\text{isotropic}(AC),\\ \neg\text{isotropic}(BC).\end{array}\right. \qquad DS_s$$

The equation part of the hypotheses is

$$\left.\begin{array}{l}\text{perpendicular}(A,B,D,G),\\ \text{perpendicular}(A,C,D,F),\\ \text{perpendicular}(B,C,D,E),\\ \text{collinear}(A,B,G),\\ \text{collinear}(A,C,F),\\ \text{collinear}(B,C,E),\\ \text{congruent}(O,A,O,B),\\ \text{congruent}(O,A,O,C),\\ \text{congruent}(O,A,O,D).\end{array}\right. \qquad HS_s$$

Nondegenerate conditions D_s are exactly what we discussed in Section 2. Then the exact statement of Simson's theorem according to the constructions in (6.1) is:

$$\forall A \cdots \forall G[HS_s \wedge DS_s \Rightarrow \text{collinear}(E,F,G)]. \tag{6.5}$$

Note that for the same theorem, the construction sequence is usually not unique. Different construction sequences lead to different non-degenerate conditions and slightly different "the exact statements" of the theorem. For example, we have at least 8 essentially different construction sequences for Simson's theorem. However, for all different construction sequences, *the equation part of the hypotheses is always the same*; in this example, it is always HS_s.

The non-degenerate conditions for the Butterfly theorem according to the construction sequence in (6.1) are

[2] Depending on the context, HS can also denote the conjunction of its elements, i.e., $HS = H_1 \wedge \cdots \wedge H_r$. The same convention is for DS and other sets of geometric conditions.

$$\begin{array}{l} \neg\text{parallel}(A,C,B,D), \\ \neg\text{perpendicular}(A,D,O,E), \\ \neg\text{parallel}(E,F,B,C). \end{array} \qquad DS_b$$

The equation part of the hypotheses is

$$\begin{array}{l} \text{congruent}(O,A,O,B), \\ \text{congruent}(O,A,O,C), \\ \text{congruent}(O,A,O,D), \\ \text{collinear}(A,E,C), \\ \text{collinear}(B,E,D), \\ \text{perpendicular}(O,E,E,F), \\ \text{collinear}(E,F,G), \\ \text{collinear}(F,A,D), \\ \text{collinear}(G,B,C). \end{array} \qquad HS_b$$

The exact statement of the Butterfly theorem according to the constructions in (6.1) is:

$$\forall A \cdots \forall G[HS_b \wedge DS_b \Rightarrow \text{midpoint}(F,E,G)]. \tag{6.6}$$

The results in Sections 4 and 5 of [4] show that either (6.5) (or (6.6)) is valid in Euclidean geometry, or it cannot be valid in Euclidean geometry no matter how many additional non-degenerate conditions are added as long as the hypotheses keep consistent.

6.5. A Method for Generation of Constructive Sequences

First we point out that the equation part of the hypothesis of a geometry statement of equality type is always easy to identify and clear cut. If the user misses one and the prover answers "not a theorem", it is user's own fault. However, if the user misses one of the necessary non-degenerate conditions and the prover answers "not a theorem", then the user is probably innocent. Even experts feel hard to deal with non-degenerate conditions (see [4] for details). In this sense, Formulation F1 is better because we don't have to concern with some very subtle degenerate cases. Besides, if the prover answers "the statement is generally false", then we know the nature of the statement: it would be useless to search for missing degenerate cases. For Class C, we even have a method for generating the inequation part DS. For a given geometry statement in Class C, the sequence of constructions is not unique. Different construction sequences generally lead to different inequation parts, thus giving slightly different exact versions of the original statement.

The equation parts of Simson's theorem and the Butterfly theorem are clear which are specified in the inputs in Section 2.

For Simson's theorem, HS_s is:

$$\begin{array}{l} \text{perpendicular}(A,B,D,G), \\ \text{perpendicular}(A,C,D,F), \\ \text{perpendicular}(B,C,D,E), \\ \text{collinear}(A,B,G), \\ \text{collinear}(A,C,F), \\ \text{collinear}(B,C,E), \\ \text{congruent}(O,A,O,B), \end{array} \qquad HS_s$$

congruent(O, A, O, C),
congruent(O, A, O, D).

and for the Butterfly Theorem, HS_b is

$$\begin{array}{l} \text{congruent}(O, A, O, B), \\ \text{congruent}(O, A, O, C), \\ \text{congruent}(O, A, O, D), \\ \text{collinear}(A, E, C), \\ \text{collinear}(B, E, D), \\ \text{perpendicular}(O, E, E, F), \\ \text{collinear}(E, F, G), \\ \text{collinear}(F, A, D), \\ \text{collinear}(G, B, C). \end{array} \qquad HS_b$$

If we know the construction sequence for Simson's theorem, then generating the inequation part DS is straightforward by the method in the previous subsection. Generally, we cannot generate construction sequence merely by the equation part HS_s. However, to ease the user for specifying the construction sequence, our prover has a method so that the user only needs to specify an order in which the points are constructed. For example for Simson's theorem, we can arrange the points in the order A, B, C, O, D, E, F, G (the first line of the input). Our heuristic to figure out the construction sequence works as follows:

Check last point (here G) to see which predicates in HS_s involve this point. If there are more than two such predicates, we simply return the answer "the order is not chosen in a proper way or the statement is not of constructive type." Otherwise, there are only one or two predicates involved. Our prover can figure out whether it is one of constructions 6–10.[3] In this case, we have two conditions, perpendicular(A, B, D, G) and collinear(A, B, G). Thus, the prover figures out that it is construction 8.3: $G = T(D, AB) \cap L(AB)$. Then we delete these two conditions from the set HS_s and go to the next point, i.e., F. Similarly, the prover finds two conditions in the new HS_s involving F and figures out the construction $F = T(D, AC) \cap L(AC)$. Next, $E = T(D, BC) \cap L(BC)$; D is on $(O, (OA))$. Last, for point O, congruent(O, A, O, B) is recognized by our prover as "O is on $B(AB)$". Thus the last construction is $O = B(AB) \cap B(AC)$. After that, HS_s is empty, thus the remaining points A, B, and C can be arbitrarily chosen. By the method in previous subsection, our prover then produces the set DS_s of non-degenerate conditions from the above construction sequence:

$\neg$collinear(A, B, C),
$\neg$isotropic(AB),
$\neg$isotropic(AC),
$\neg$isotropic(BC).

Then the exact statement of Simson's theorem is:

$$\forall A \cdots \forall G[HS_s \wedge DS_s \Rightarrow \text{collinear}(E, F, G)]. \tag{3.3}$$

Now let us look at the Butterfly Theorem. If we arrange the points in the order O, A, B, C, D, E, F, G, then we have the construction sequence:

[3] As we mentioned before, the prover can figure out more than those constructions. But for the purpose of discussion of basic techniques and principles, constructions 6–10 are enough.

O and A are arbitrarily chosen;	construction 1
B is on $(O, (OA))$;	construction 7
C is on $(O, (OA))$;	construction 7
D is on $(O, (OA))$;	construction 7
$E = L(AC) \cap L(BD)$;	construction 8.1
$F = L(AD) \cap T(E, OE)$;	construction 8.3
$G = L(EF) \cap L(BC)$.	construction 8.1

Then the program generates a set DS_b of non-degenerate conditions from the above construction sequence:

$\neg$parallel(E, F, B, C),
$\neg$perpendicular(A, D, O, E),
$\neg$parallel(A, C, B, D).

The exact statement of the Butterfly Theorem is:

$$\forall A \cdots \forall G[HS_b \wedge DS_b \Rightarrow \text{midpoint}(F, E, G)]. \tag{3.4}$$

Note that for the same theorem, the construction sequence is usually not unique. Different construction sequences lead to different non-degenerate conditions and slightly different "the exact statements" of the theorem. For example, we have at least 8 essentially different construction sequences for Simson's theorem (see Appendix 2 of [4]).