文章编号:1674-7070(2009)02-0134-08

关于 Steiner-Lehmes 定理的 4 个推广猜想的研究

廖文访[1]　许明春[1]

摘要

著名的 Steiner-Lehmes 定理旨在一定条件下判断 1 个三角形是否为等腰三角形,而王彦海把题设条件替换成了各种形式,并提出了 9 个猜想. 用吴方法和 MMP 软件证明了其中的 3 个,提出并解决 1 个新的猜想. 余下的几个猜想由于题设条件转化成代数条件的多项式次数较高,MMP 软件运行时间太长而未能得到期望的结果.

关键词

Steiner-Lehmes 定理;吴方法;MMP 软件

中图分类号 G633.62

文献标志码 A

0　引言

Introduction

对于著名的 Steiner-Lehmes 定理(三角形 2 角的角平分线长相等,则三角形是等腰三角形),文献[1]把题设条件替换成了各种形式,提出了 9 个猜想,并肯定地证明了前 3 个. 本文运用吴方法和 MMP(Mathematics Mechanization Platform)软件在计算机上证明了下面 11 个猜想中的 3 个,在此基础上得到新的猜想并给出了证明. 现在讨论关于 Steiner-Lehmes 定理若干推广形式的问题.

问题 1　如图 1,在三角形 ABC 中,D、E 分别为 AC、AB 反向延长线的点,BD、CE 相交于 P,且 $BD = CE$,当 P 满足下列条件之一时,则 $AB = AC$.

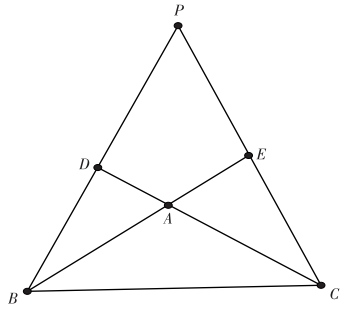

图 1

Fig. 1

1) AP 为 $\angle BPC$ 的平分线;

2) P 在 BC 中垂线上;

3) P 在 BC 中线的方向延长线上;

4)(猜想)AP 为 $\angle BAC$ 平分线的反向延长线;

5)(猜想)P 在 BC 高线的反向延长线上;

6)(猜想)$\angle ABP - \angle ACP = k(\angle PBC - \angle PCB)$($k$ 为常数);

7)(猜想)$AB \times PB = AC \times PC$;

8)(猜想)$AB \times PD = AC \times PE$;

9)(猜想)$AB + (-)PB = AC + (-)PC$;

10)(猜想)$AB + (-)QB = AC + (-)QC$(Q 为 PA 的延长线与 BC 的交点);

收稿日期 2009-07-07

资助项目 国家自然科学基金(10771077)

作者简介

廖文访,男,硕士生,从事代数学研究. fangliaowen@163.com

许明春(通讯作者),男,理学博士,副教授,硕士生导师,从事代数学研究. xumingchun2002@yahoo.com.cn

1　华南师范大学　数学科学学院,广州,510620

11)(猜想)$AD+(-)PD=AE+(-)PE$;

12)(作者提出的猜想)$AB^2+PB^2=AC^2+PC^2$.

其中问题的情形 1)~3)已在文献[1]中被证明,本文将给出问题的猜想 4),5),7)和 12)的证明.

先简单介绍一下吴方法的背景.20 世纪 70 年代末,吴文俊先生受中国古代数学机械化思想影响,借助 20 世纪 30 年代 Ritt 的理论工作,针对几何定理机器证明问题研究和发展得出新方法——吴方法.吴先生不仅做了许多理论工作,而且进行了大量的机证实验,发现并证明了不少几何定理,将几何机械化乃至整个数学的进程向前推进了一大步.适用吴方法证明的定理具有假设与结论部分的代数关系式都可以用多项式方程来表示的特征.

1 算法原理

Principle of the algorithm

为了接下来的应用,下面简要说明吴方法的算法原理,并引出其中的数学理论.

1.1 吴方法的证明例子

下面通过中位线定理说明吴方法的证明过程和算法原理.

例 如图 2,在△ABC 中,D 是 AB 的中点,$DE/\!/BC$ 交 AC 于 E,求证:E 是 AC 的中点.

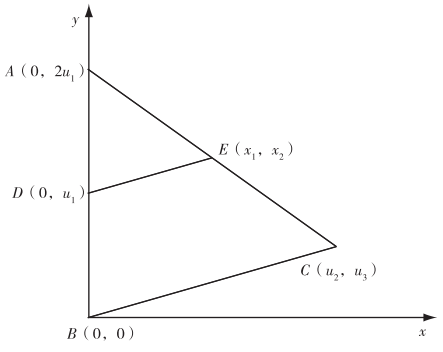

图 2

Fig. 2

证 首先选取坐标系,以△ABC 的 AB 边为 y 轴,点 B 为原点,则可设 $A(0,2u_1)$,$B(0,0)$,$C(u_2,u_3)$,$D(0,u_1)$,$E(x_1,x_2)$,其中 u_1,u_2,u_3 为自由变元,x_1,x_2 为约束变元.哪些点是自由点,哪些是独立点,往往不是固定的.如果取某些点为独立点,那么其他点就不是独立点.但在同一个问题中,独立点的个数则是确定的.吴方法证明如下:

第 1 步 将几何问题化为代数形式.

假设部分,由 $DE/\!/BC$,有

$$f_1=(x_2-u_1)(u_2-0)-(x_1-0)(u_3-0)=u_2x_2-u_3x_1-u_1u_2=0,$$

再由 A,E,C 3 点共线,得

$$f_2=(x_2-2u_1)(u_2-0)-(x_1-0)(u_3-2u_1)=u_2x_2-u_3x_1+2u_1x_1-2u_1u_2=0.$$

结论部分:要证 $AE=EC$,即

$$g=x_1^2+(x_2-2u_1)^2-(u_2-x_1)^2-(u_3-x_2)^2=-4u_1x_2+2u_3x_2+2u_2x_1+4u_1^2-u_2^2-u_3^2=0.$$

第 2 步 整序—三角化—吴升列.

求出 f_1,f_2 的开列 f_1^*,f_2^*,可调用 wsolve 函数时得到升列.以本例而论,A,B,C,D 4 点的位置定了,其它点的位置也就定了,而 A,C,D 3 点的位置可由 3 个坐标 u_1,u_2,u_3 确定,这样在 u_1,u_2,u_3,x_1,x_2 的 5 个变元中,只有 3 个是自由的,另外 2 个是受约束的.本例可手算得到.整序就是把约束变元排顺序,使得第 1 个约束变元直接跟着自由变元走,第 $k+1$ 个约束变元直接跟着自由变元和前 k 个约束变元走,也就是说把假设条件改写成 1 组等式:$f_1^*=f_2^*=\cdots=f_k^*=0$.其中 f_1^* 中只出现自由变元和第 1 个约束变元,f_k^* 中只出现自由变元和前 k 个约束元.

令:

$$f_1^*=f_2-f_1=2u_1x_1-u_1u_2=0,$$

$$f_2^*=f_2=u_2x_2-u_3x_1+2u_1x_1-2u_1u_2=0.$$

第 3 步 做逐步除法—验证结论真伪.

将 g 除以 f_2^*,并将 g 和 f_2^* 都看作约束变元 x_2 的多项式,为了避免商式中出现分式,实际上是将 u_2g 除以 f_2^*,得

$$u_2g=2(u_3-2u_1)f_2^*+R_2. \tag{1}$$

其中 $R_2=2(u_2^2+u_3^2-4u_1u_3+4u_1^2)x_1-4u_1^2u_2-u_2^3-u_2u_3^2+4u_1u_2u_3$.

再将 R_2 除以 f_1^*,把它们都看作 x_1 的多项式,为了避免商式中出现分式,实际上是将 u_1R_2 除以 f_1^*,即:

$$u_1R_2=(u_1^2+u_3^2-4u_1u_2+4u_1^2)f_1^*+R. \tag{2}$$

将式(1)乘以 u_1,再将式(2)代入,得

$$u_1u_2g=2u_1(u_3-2u_1)f_2^*+(u_1^2+u_3^2-4u_1u_2+4u_1^2)f_1^*+R.$$

当 $u_1\neq0,u_2\neq0$,由 $f_2^*=0,f_1^*=0,R=0$,必得 $g=0$,命题得证.

其中 $u_1\neq0,u_2\neq0$ 为非退化条件.当 $u_1=0$,点 A 与点 B 重合,ABC 不成为三角形.当 $u_2=0$ 时,则点

C 在 AB 边上,因此此命题在 $u_1 \neq 0, u_2 \neq 0$ 的条件下才能成立. 现在用 MMP 软件运算.

屏幕显示如下内容:

```
f1:= u2 * x2 - u3 * x1 - u1 * u2;
f1:= - x1 * u3 - u2 * u1 + u2 * x2
f2:= u2 * x2 - u3 * x1 + 2 * u1 * x1 - 2 * u1 * u2;
f2:= u2 * x2 - u3 * x1 + 2 * u1 * x1 - 2 * u1 * u2
g:= - 4 * u1 * x2 + 2 * u3 * x2 + 2 * u2 * x1 + 4 * u1^2 - u2^2 - u3^2;
g:= - 4 * u1 * x2 + 2 * u3 * x2 + 2 * u2 * x1 + 4 * u1^2 - u2^2 - u3^2
wsolve([f1,f2],[x1,x2],[]);
                    /调用 wsolve 函数求吴升列/
[[u2 - 2 * x1, 2 * x2 - 2 * u1 - u3]]
                    /得到一组吴升列/
premas(g,[u2 - 2 * x1, 2 * x2 - 2 * u1 - u3],[x1,x2]);
                    /调用 premas 函数做逐步除法/
0
init(u2 - 2 * x1,[x1,x2]);
                    /调用 init 函数求初式/
 - 2                /初式不为 0,条件非退化/
init(2 * x2 - 2 * u1 - u3,[x1,x2]);
                    /调用 init 函数求初式/
2                   /初式不为 0,条件非退化/
```

吴方法的证明过程类似于解线性方程组中的消元法,只是其消元过程是通过伪除实现的,其目的是实现多项式的方程组的三角化.

用传统方法证明每 1 个稍难的初等几何命题,都要经过一番巧思. 有的要添这样的辅助线,有的要添那样的辅助线. 不同的命题有不同的证法,也就需要不同的巧思. 1 个新的命题需要通过新的巧思,找到新的证法. 而吴方法不是特殊地适用于个别的命题,而是普遍地适用于初等几何的所有命题,至少是某 1 类的很多命题. 只要按照这种方法机械地进行,在有限步之后,就可对这一类中的任何初等几何命题判定它是真是假. 吴方法只需机械地进行,对于这类中的任何命题都是按照同样的步骤进行,不必对特殊的命题运用特殊的巧思,这正是吴方法的优胜之处.

通过上述的例题的证明过程和算法原理,下面简要分析其中引出的数学原理.

1.2 吴方法的原理

首先介绍 1 个定义.

定义[4] 设 K 为数域,特征为 0,如果多项式组 $A_1, A_2, \cdots, A_n$ 包含于多项式环 $K[x_1, \cdots, x_n, u_1, \cdots, u_r]$ ($x_1, x_2, \cdots, x_n$ 称为约束变元;$u_1, u_2, \cdots, u_r$ 称为自由变元),并满足条件:对任 1 多项式 A_i,A_i 中包含的变元的最大下标为 i,即 A_i 可按 x_i 的幂次写为

$$A_i(x_1, \cdots, x_i, u_1, \cdots, u_r) = I_i x_i^{M_i} + x_i$$

的低次项,其中 I_i 为 $K[x_1, \cdots, x_{i-1}, u_1, \cdots, u_r]$ 中的多项式,称为 A_i 的初式.

定理 1[4] 设 $F, G \in K[x_1, \cdots, x_n]$,$G$ 对于 F 未约化,G 除 F 的余式为 R,则 $V(F,G) = V(F,G,R)$. 更进一步,如果初式 $I(F)$ 在 $V(F,G)$ 上不为 0,则 $V(F,G) = V(F,R)$.

在上面例题证明的第 1 步中,吴方法是基于定理 1 的,通过将几何条件转化为代数条件,在初式不为 0 的条件下,这种转化是等价的.

定理 2(Ritt 原理)[4] 任给多项式组 PS,可以机械地得到 1 个三角化多项式组 TPS(不唯一,称 PS 的特征组),使

$$V(TPS/J) \subset V(PS) \subset V(TPS),$$

$$V(PS) = V(TPS/J) + \sum V(PS_i).$$

其中:J 指 TPS 中诸初式 I_i 的乘积;PS_i 指将 I_i 添入 PS 后的多项式组.

定理 3[4] 给定多项式组 $PS = \{g_1, \cdots, g_s\} \subset K[x_1, \cdots, x_n]$,存在 1 种机械化算法,经过有限步运算后,可以由 PS 得到 1 个基列,使得 PS 中每个多项式对此基列求余所得的余式为 0,如果这个基列是非矛盾的,即为 PS 的特征列.

接着,在第 2 步中吴方法利用了著名 Ritt 定理和定理 3,通过整序得到三角升列,也就是吴升列.

定理 4[4] 设 AS 是 $K[x_1, \cdots, x_n]$ 中 1 组多项式的升列 A_i,其初式为 I_i,任给多项式 G,不论 AS 是否可约,在非退化条件 $I_i \neq 0, i = 1, \cdots, n$ 下,当 $R = 0$ 时,$G = 0$ 可以由 $A_i = 0$ 推出.

最后,在第 3 步中借助于定理 4,吴方法通过对三角升列做逐步除法来验证命题的真伪.

2 吴方法的应用

Application of Wu-method

现在就利用吴方法和 MMP 软件来讨论引言提出的问题中的 4 个猜想.

问题形式 4 如图 3,首先选取坐标系,取 BC 的长为 2(显然这种取法是合理的),各点坐标如图所示. 下面将题设的几何条件化成代数条件:

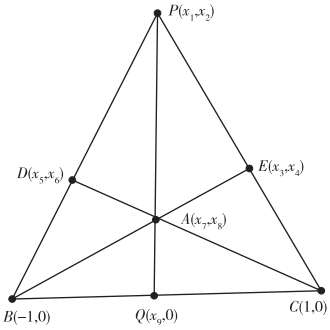

图 3
Fig. 3

点 P、D、B 共线⇒

$$f_1 = x_6 - x_2 + x_1x_6 - x_5x_2 = 0; \quad (3)$$

点 P、E、C 共线⇒

$$f_2 = -x_4 + x_2 + x_1x_4 - x_3x_2 = 0; \quad (4)$$

点 B、A、E 共线⇒

$$f_3 = x_7x_4 - x_8x_3 + x_4 - x_8 = 0; \quad (5)$$

点 C、A、D 共线⇒

$$f_4 = x_7x_6 - x_8x_5 - x_6 + x_8 = 0; \quad (6)$$

点 P、A、Q 共线⇒

$$f_5 = -x_8x_9 + x_9x_2 + x_1x_8 - x_7x_2; \quad (7)$$

PQ 为 $\angle BAC$ 的平分线⇒

$$f_6 = -4x_7^2x_9 + 4x_7x_9^2 + 4x_7 - 4x_9 - 4x_8^2x_9 = 0\left(\frac{AB}{BQ} = \frac{AC}{CQ}\right); \quad (8)$$

$BD = CE$⇒

$$f_7 = x_5^2 + 2x_5 + x_6^2 - x_3^2 + 2x_3 - x_4^2 = 0; \quad (9)$$

要证结论

$$AB = AC \Leftrightarrow f_8 = x_7 = 0. \quad (10)$$

其中:x_2,x_8 作为自由变元;x_1,x_3,x_4,x_5,x_6,x_7,x_9 作为约束变元.

现在用 MMP 验证结论,其过程与例题类似,但因为运算的程序需要比较多的篇幅,所以具体程序附在本文的末尾供参考,在这里只对程序的结果进行分析. 在调用 wsolve 函数时得到 2 组升列,接着分别调用 premas 函数做逐步除法得到第 2 组不为 0,所以可以不予考虑. 第 1 组为 0,这是所期望的结果,现在讨论第 1 组的初式是否含有退化的情况(即初式为 0). 通过调用 init 函数在程序中得到的初式分别为 1 和 $x_2 + x_8$,显然由图 3 可知 $x_2 + x_8 \neq 0$,当然 $1 \neq 0$. 因此本组非退化,故 $x_7 = 0$,猜想得证.

评注:通过观察可以知道,这里不仅证明了 D、E 分别为 AC、AB 反向延长线的点的情况,而且证明了 D、E 分别为 AC、AB 延长线的点的情况也成立的. 这也是吴方法突出的优点,不但可以判断原命题的正确与否,还可以在这个过程中发现新的性质. 这一优点在接着下来的证明中同样可以体现出来.

问题形式 5 如图 4,与问题形式 4 同理,根据题意取各点坐标如图. 下面将题设的几何条件化成代数条件:

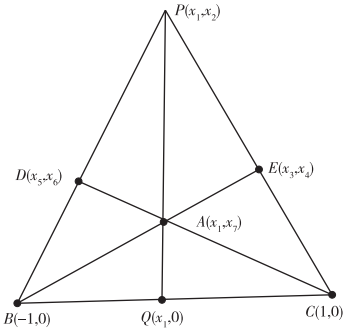

图 4
Fig. 4

点 P、D、B 共线⇒

$$f_1 = x_6 - x_2 + x_1x_6 - x_5x_2 = 0; \quad (11)$$

点 P、E、C 共线⇒

$$f_2 = -x_4 + x_2 + x_1x_4 - x_3x_2 = 0; \quad (12)$$

点 B、A、E 共线⇒

$$f_3 = x_1x_4 - x_7x_3 + x_4 - x_7 = 0; \quad (13)$$

点 C、A、D 共线⇒

$$f_4 = x_1x_6 - x_7x_5 - x_6 + x_7 = 0; \quad (14)$$

$BD = CE$⇒

$$f_7 = x_5^2 + 2x_5 + x_6^2 - x_3^2 + 2x_3 - x_4^2 = 0; \quad (15)$$

要证结论

$$AB = AC \Leftrightarrow f_6 = x_1 = 0. \quad (16)$$

其中:x_2,x_7 作为自由变元;x_1,x_3,x_4,x_5,x_6 作为约束变元.

与问题形式 4 证明一样,本文只对程序的结果进行分析. 在调用 wsolve 函数时得到 2 组升列,接着分别调用 premas 函数做逐步除法得到第 2 组不为 0,现在讨论第 1 组的初式是否含有退化的情况(即初式为 0). 通过调用 init 函数在程序中得到的初式分别为 1 和 $x_2 + x_7$,显然由图 4,可知 $x_2 + x_7 \neq 0$,当然 $1 \neq 0$. 因此本组非退化,故 $x_1 = 0$,猜想得证.

如问题形式 4 评注,D、E 分别为 AC、AB 延长线的点的情况也成立的.

问题形式 7 如图 5,与问题形式 4 同理,根据题意取各点坐标如图. 下面将题设的几何条件化成

代数条件:

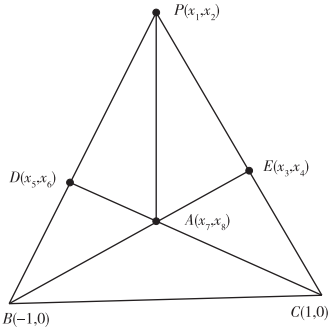

图 5
Fig. 5

点 P、D、B 共线⇒

$$f_1 = x_6 - x_2 + x_1x_6 - x_5x_2 = 0; \quad (17)$$

点 P、E、C 共线⇒

$$f_2 = -x_4 + x_2 + x_1x_4 - x_3x_2 = 0; \quad (18)$$

点 B、A、E 共线⇒

$$f_3 = x_7x_4 - x_8x_3 + x_4 - x_8 = 0; \quad (19)$$

点 C、A、D 共线 ⇒

$$f_4 = x_7x_6 - x_8x_5 - x_6 + x_8 = 0; \quad (20)$$

$BD = CE$⇒

$$f_7 = x_5^2 + 2x_5 + x_6^2 - x_3^2 + 2x_3 - x_4^2 = 0; \quad (21)$$

$AB \times PB = AC \times PC$⇒

$$f_6 = 4x_1 + 4x_7 + 4x_7^2x_1 + 4x_7x_1^2 + 4x_7x_2^2 + 4x_8^2x_1 = 0; \quad (22)$$

要证结论

$$AB = AC \Leftrightarrow f_7 = x_7 = 0. \quad (23)$$

其中:x_2,x_8 作为自由变元;x_1,x_3,x_4,x_5,x_6,x_7 作为约束变元.

与问题形式 4 证明一样,本文只对程序的结果进行分析. 先调用 wsolve 函数时得到 1 组升列,再调用 premas 函数做逐步除法时结果为 0. 现在讨论初式是否含有退化的情况(即初式为 0). 通过调用 init 函数在程序中得到的初式分别为 1 和 $x_2 + x_8$,显然由图 5 可知 $x_2 + x_8 \neq 0$,当然 $1 \neq 0$. 因此本组非退化,故 $x_7 = 0$,猜想得证.

如问题形式 4 评注,D、E 分别为 AC、AB 延长线的点的情况也是成立的.

问题形式 12 如图 5,与问题形式 4 同理,根据题意取各点坐标如图. 下面将题设的几何条件化成代数条件:

点 P、D、B 共线⇒

$$f_1 = x_6 - x_2 + x_1x_6 - x_5x_2 = 0; \quad (24)$$

点 P、E、C 共线⇒

$$f_2 = -x_4 + x_2 + x_1x_4 - x_3x_2 = 0; \quad (25)$$

点 B、A、E 共线⇒

$$f_3 = x_7x_4 - x_8x_3 + x_4 - x_8 = 0; \quad (26)$$

点 C、A、D 共线⇒

$$f_4 = x_7x_6 - x_8x_5 - x_6 + x_8 = 0; \quad (27)$$

$BD = CE$⇒

$$f_7 = x_5^2 + 2x_5 + x_6^2 - x_3^2 + 2x_3 - x_4^2 = 0; \quad (28)$$

$AB^2 + PB^2 = AC^2 + PC^2$⇒

$$f_6 = 4x_7^2x_1 + 4x_7x_1^2 + 4x_7x_2^2 + 4x_8^2x_1 + 4x_1 + 4x_7 = 0; \quad (29)$$

要证结论

$$AB = AC \Leftrightarrow f_7 = x_7 = 0. \quad (30)$$

其中:x_2,x_8 作为自由变元;x_1,x_3,x_4,x_5,x_6,x_7 作为约束变元.

与问题形式 4 证明一样,本文只对程序的结果进行分析. 先调用 wsolve 函数时得到 1 组升列,再调用 premas 函数做逐步除法时结果为 0. 现在讨论初式是否含有退化的情况(即初式为 0). 通过调用 init 函数在程序中得到的初式分别为 1 和 $x_2 + x_8$,显然由图 5 可知 $x_2 + x_8 \neq 0$,当然 $1 \neq 0$. 因此本组非退化,故 $x_7 = 0$,猜想得证.

如问题形式 4 评注,D、E 分别为 AC、AB 延长线的点的情况也成立的.

3 结束语
Concluding remarks

吴方法应用于平面几何的证明,是可以利用软件 MMP 来进行证明出来的. 它有一定的步骤,只要建立适当的平面直角坐标系,正确的将命题中的已知,求证当中所包含的几何关系表示为代数的形式,然后输入计算机,调用函数 wsolve,premas 和 init,就能在短短的几秒钟求解出来,这比利用传统的初等方法去证明节省更多的时间和精力,显示了吴方法的优越性. 无可否认吴方法在理论上是完美的,但在余下的几个猜想中由于题设条件转化成代数条件的多项式次数较高,MMP 软件运行时间太长而未能得到期望的结果.

参考文献
References

[1] 王彦海. 关于 Steiner-Lehmes 定理外等长分角线的猜想的研究[J]. 乌鲁木齐成人教育学院学报,2004,12(3):82-85
WANG Yanhai. Researchs on the development of Steiner-Lehmes theorem[J]. Journal of Urumqi Adult Education Institute,2004,12(3):82-85

[2] 吴文俊. 王者之路[M]. 长沙:湖南科学技术出版社,1999:

28-61
WU Wenjun. The road of king[M]. Changsha:Hunan Science & Technology Press,1999:28-61
[3] 高小山. MMP 软件与软件使用手册[Z]. 北京:中国科学院数学机械化重点实验室,2006:57-61
GAO Xiaoshan. MMP software and its manual[Z]. Beijing:Key Lab for Mechanization of Mathematics,CAS,2006:57-61
[4] 吴文俊. 吴文俊论数学机械化[M]. 济南:山东教育出版社,1995:358-499
WU Wenjun. WU Wenjun's theory about mechanization of mathematics[M]. Jinan:Shandong Education Press,1995:358-499

附录:问题形式 4、5、7、12 的程序

问题形式 4 的程序

```
f1:=x6-x2+x1*x6-x5*x2
f1:=-x2*x5-x2+x6*x1+x6;
f2:=-x4+x2+x1*x4-x3*x2;
f2:=-x4+x1*x4-x2*x3+x2
f3:=x7*x4-x8*x3+x4-x8;
f3:=-x8-x3*x8+x4*x7+x4
f4:=x7*x6-x8*x5-x6+x8;
f4:=x7*x6-x8*x5-x6+x8
f5:=-x8*x9+x9*x2+x1*x8-x7*x2;
f5:=-x8*x9+x9*x2+x1*x8-x7*x2
f6:=-4*x7^2*x9+4*x7*x9^2+4*x7-4*x9-4*x8^2*x9;
f6:=-4*x7^2*x9+4*x7*x9^2+4*x7-4*x9-4*x8^2*x9
f7:=x5^2+2*x5+x6^2-x3^2+2*x3-x4^2;
f7:=x5^2+2*x5+x6^2-x3^2+2*x3-x4^2
wsolve([f1,f2,f3,f4,f5,f6,f7],[x1,x3,x4,x5,x6,x7,x9],[]);
[[x1,x3*x2-x2+x8*x3+x8,x4*x2-2*x8*x2+x8*x4,x5*x2+x2+x8*x5-x8,x2*x6+x8*x6-2*x8*x2,x7,x9],[x9^2*x1*x2^3-x1*x2^3-x9^3*x2^3+x9*x2^3-3*x9^2*x8*x1*x2^2-x8*x1*x2^2+x9^3*x8*x2^2-x9*x8*x2^2+x9^2*x8^2*x1*x2-x8^2*x1*x2+x9^3*x8^2*x2-x9*x8^2*x2+x9^2*x8^3*x1-x8^3*x1-x9^3*x8^3+x9*x8^3,x9^3*x3*x2^4+x9^2*x3*x2^4-x9*x3*x2^4-x3*x2^4-x9^3*x2^4-x9^2*x2^4+x9*x2^4+x2^4-4*x9^3*x8*x3*x2^3-2*x9^2*x8*x3*x2^3-2*x8*x3*x2^3+2*x9^3*x8*x2^3+4*x9^2*x8*x2^3+2*x9*x8*x2^3+4*x9^3*x8^2*x3*x2^2-2*x9^2*x8^2*x3*x2^2-2*x8^2*x3*x2^2-2*x9^3*x8^2*x2^2-4*x9^2*x8^2*x2^2-2*x9*x8^2*x2^2+2*x9^2*x8^3*x3*x2-2*x8^3*x3*x2+2*x9^3*x8^3*x2-2*x9*x8^3*x2-x9^3*x8^4*x3+x9^2*x8^4*x3+x9*x8^4*x3-x8^4*x3-x9^3*x8^4+x9^2*x8^4+x9*x8^4-x8^4,x9*x4*x2+x4*x2-2*x8*x2-x9*x8*x4+x8*x4,x9^3*x5*x2^4-x9^2*x5*x2^4-x9*x5*x2^4+x5*x2^4+x9^3*x2^4-x9^2*x2^4-x9*x2^4+x2^4-4*x9^3*x8*x5*x2^3+2*x9^2*x8*x5*x2^3+2*x8*x5*x2^3-2*x9^3*x8*x2^3+4*x9^2*x8*x2^3-2*x9*x8*x2^3+4*x9^3*x8^2*x5*x2^2+2*x9^2*x8^2*x5*x2^2+2*x8^2*x5*x2^2+2*x9^3*x8^2*x2^2-4*x9^2*x8^2*x2^2+2*x9*x8^2*x2^2-2*x9^2*x8^3*x5*x2+2*x8^3*x5*x2-2*x9^3*x8^3*x2+2*x9*x8^3*x2-x9^3*x8^4*x5-x9^2*x8^4*x5+x9*x8^4*x5+x8^4*x5+x9^3*x8^4+x9^2*x8^4-x9*x8^4-x8^4,x9*x2*x6-x2*x6-x9*x8*x6-x8*x6+2*x8*x2,x9^2*x7*x2^3-x7*x2^3-x9^3*x2^3+x9*x2^3-3*x9^2*x8*x7*x2^2-x8*x7*x2^2+3*x9^3*x8*x2^2+x9*x8*x2^2+x9^2*x8^2*x7*x2-x8^2*x7*x2-3*x9^3*x8^2*x2-x9*x8^2*x2+x9^2*x8^3*x7-x8^3*x7+x9^3*x8^3-x9*x8^3,x9^4*x2^6-2*x9^2*x2^6+x2^6-6*x9^4*x8*x2^5+4*x9^2*x8*x2^5+2*x8*x2^5+11*x9^4*x8^2*x2^4+2*x9^2*x8^2*x2^4+3*x8^2*x2^4+2*x9^6*x2^4-2*x9^4*x2^4-2*x9^2*x2^4+2*x2^4-4*x9^4*x8^3*x2^3+4*x8^3*x2^3-8*x9^6*x8*x2^3+4*x9^4*x8*x2^3+4*x8*x2^3-5*x9^4*x8^4*x2^2+2*x9^2*x8^4*x2^2+3*x8^4*x2^2+12*x9^6*x8^2*x2^2-4*x9^4*x8^2*x2^2+4*x9^2*x8^2*x2^2+4*x8^2*x2^2+2*x9^4*x8^5*x2-4*x9^2*x8^5*x2+2*x8^5*x2-8*x9^6*x8^3*x2+4*x9^4*x8^3*x2+4*x8^3*x2+x9^4*x8^6-2*x9^2*x8^6+x8^6+2*x9^6*x8^4-2*x9^4*x8^4-2*x9^2*x8^4+2*x8^4]]
premas(x7,[x1,x3*x2-x2+x8*x3+x8,x4*x2-2*x8*x2+x8*x4,x5*x2+x2+x8*x5-x8,x2*x6+x8*x6-2*x8*x2,x7,x9],[x1,x3,x4,x5,x6,x7,x9]);
0
premas(x7,[x9^2*x1*x2^3-x1*x2^3-x9^3*x2^3+x9*x2^3-3*x9^2*x8*x1*x2^2-x8*x1*x2^2+x9^3*x8*x2^2-x9*x8*x2^2+x9^2*x8^2*x1*x2-x8^2*x1*x2+x9^3*x8^2*x2-x9*x8^2*x2+x9^2*x8^3*x1-x8^3*x1-x9^3*x8^3+x9*x8^3,x9^3*x3*x2^4+x9^2*x3*x2^4-x9*x3*x2^4-x3*x2^4-x9^3*x2^4-x9^2*x2^4+x9*x2^4+x2^4-4*x9^3*x8*x3*x2^3-2*x9^2*x8*x3*x2^3-2*x8*x3*x2^3+2*x9^3*x8*x2^3+4*x9^2*x8*x2^3+2*x9*x8*x2^3+4*x9^3*x8^2*x3*x2^2-2*x9^2*x8^2*x3*x2^2-2*x8^2*x3*x2^2-2*x9^3*x8^2*x2^2-4*x9^2*x8^2*x2^2-2*x9*x8^2*x2^2+2*x9^2*x8^3*x3*x2-2*x8^3*x3*x2+2*x9^3*x8^3*x2-2*x9*x8^3*x2-x9^3*x8^4*x3+x9^2*x8^4*x3+x9*x8^4*x3-x8^4*x3-x9^3*x8^4+x9^2*x8^4+x9*x8^4-x8^4,x9*x2*x4+x2*x4-x9*x8*x4+x8*x4-2*x8*x2,x9^3*x5*x2^4-x9^2*x5*x2^4-x9*x5*x2^4+x5*x2^4+x9^3*x2^4-x9^2*x2^4-x9*x2^4+x2^4-4*x9^3*x8*x5*x2^3+2*x9^2*x8*x5*x2^3+2*x8*x5*x2^3-2*x9^3*x8*x2^3+4*x9^2*x8*x2^3-2*x9*x8*x2^3+4*x9^3*x8^2*x5*x2^2+2*x9^2*x8^2*x5*x2^2+2*x8^2*x5*x2^2+2*x9^3*x8^2*x2^2-4*x9^2*x8^2*x2^2+2*x9*x8^2*x2^2-2*x9^2*x8^3*x5*x2+2*x8^3*x5*x2-2*x9^3*x8^3*x2+2*x9*x8^3*x2-x9^3*x8^4*x5-x9^2*x8^4*x5+x9*x8^4*x5+x8^4*x5+x9^3*x8^4+x9^2*x8^4-x9*x8^4-x8^4,x9*x6*x2-x6*x2+2*x8*x2-x9*x8*x6-x8*x6,x9^2*x7*x2^3-x7*x2^3-x9^3*x2^3+x9*x2^3-3*x9^2*x8*x7*x2^2-x8*x7*x2^2+3*x9^3*x8*x2^2+x9*x8*x2^2+x9^2*x8^2*x7*x2-x8^2*x7*x2-3*x9^3*x8^2*x2-x9*x8^2*x2+x9^2*x8^3*x7-x8^3*x7+x9^3*x8^3-x9*x8^3,x9^4*x2^6-2*x9^2*x2^6+x2^6-6*x9^4*x8*x2^5+4*x9^2*x8*x2^5+2*x8*x2^5+11*x9^4*x8^2*x2^4+2*x9^2*x8^2*x2^4+3*x8^2*x2^4+2*x9^6*x2^4-2*x9^4*x2^4-2*x9^2*x2^4+2*x2^4-4*x9^4*x8^3*x2^3+4*x8^3*x2^3-8*x9^6*x8*x2^3+4*x9^4*x8*x2^3+4*x8*x2^3-5*x9^4*x8^4*x2^2+2*x9^2*x8^4*x2^2+3*x8^4*x2^2+12*x9^6*x8^2*x2^2-4*x9^4*x8^2*x2^2+4*x9^2*x8^2*x2^2+4*x8^2*x2^2+2*x9^4*x8^5*x2-4*x9^2*x8^5*x2+2*x8^5*x2-8*x9^6*x8^3*x2+4*x9^4*x8^3*x2+4*x8^3*x2+x9^4*x8^6-2*x9^2*x8^6+x8^6+2*x9^6*x8^4-2*x9^4*x8^4-2*x9^2*x8^4+2*x8^4],[x1,x3,x4,x5,x6,x7,x9]);
```

```
 * x8^4 * x2^2 + 3 * x8^4 * x2^2 + 12 * x9^6 * x8^2 * x2^2 - 4 * x9^4 * x8^
2 * x2^2 + 4 * x9^2 * x8^2 * x2^2 + 4 * x8^2 * x2^2 + 2 * x9^4 * x8^5 * x2
 - 4 * x9^2 * x8^5 * x2 + 2 * x8^5 * x2 - 8 * x9^6 * x8^3 * x2 + 4 * x9^4 *
x8^3 * x2 + 4 * x8^3 * x2 + x9^4 * x8^6 - 2 * x9^2 * x8^6 + x8^6 + 2 * x9^6
 * x8^4 - 2 * x9^4 * x8^4 - 2 * x9^2 * x8^4 + 2 * x8^4 ], [ x1, x3, x4, x5,
x6, x7, x9 ]);
x2^3 * x9^3 - 3 * x8 * x2^2 * x9^3 + 3 * x8^2 * x2 * x9^3 - x8^3 * x9^3 -
x2^3 * x9 - x8 * x2^2 * x9 + x8^2 * x2 * x9 + x8^3 * x9
init( x1, [ x1, x3, x4, x5, x6, x7, x9 ]);
1
init( x3 * x2 - x2 + x8 * x3 + x8, [ x1, x3, x4, x5, x6, x7, x9 ]);
x1 + x8
init( x4 * x2 - 2 * x8 * x2 + x8 * x4, [ x1, x3, x4, x5, x6, x7, x9 ]);
x2 + x8
init( x5 * x2 + x2 + x8 * x5 - x8, [ x1, x3, x4, x5, x6, x7, x9 ]);
x2 + x8
init( x6 * x2 - 2 * x8 * x2 + x8 * x6, [ x1, x3, x4, x5, x6, x7, x9 ]);
x2 + x8
init( x7, [ x1, x3, x4, x5, x6, x7, x9 ]);
x2 + x8
init( x9, [ x1, x3, x4, x5, x6, x7, x9 ]);
1
```

问题形式 5 的程序

```
f1:= x6 - x2 + x1 * x6 - x5 * x2;
f1:= x6 - x2 + x1 * x6 - x5 * x2
f2:= - x4 + x2 + x1 * x4 - x3 * x2;
f2:= - x4 + x2 + x1 * x4 - x3 * x2
f3:= x1 * x4 - x7 * x3 + x4 - x7;
f3:= x1 * x4 - x7 * x3 + x4 - x7
f4:= x1 * x6 - x7 * x5 - x6 + x7;
f4:= x1 * x6 - x7 * x5 - x6 + x7
f5:= x5^2 + 2 * x5 + x6^2 - x3^2 + 2 * x3 - x4^2;
f5:= x5^2 + 2 * x5 + x6^2 - x3^2 + 2 * x3 - x4^2
wsolve( [ f1, f2, f3, f4, f5 ], [ x1, x3, x4, x5, x6 ], [ ]);
[[ x1, x3 * x2 - x2 + x7 * x3 + x7, x4 * x2 - 2 * x7 * x2 + x7 * x4, x5 *
x2 + x2 + x7 * x5 - x7, x2 * x6 + x7 * x6 - 2 * x7 * x2 ], [ x1 * x2 * x6 -
x2 * x6 - x7 * x1 * x6 - x7 * x6 + 2 * x7 * x2, x3 * x2^2 * x6 - x2^2 * x6
 - x7^2 * x3 * x6 - x7^2 * x6 - x7 * x3 * x2^2 + x7 * x2^2 + x7^2 * x3 * x2
 + x7^2 * x2, x4 * x2 * x6 - x7 * x2 * x6 + x7 * x4 * x6 - x7 * x4 * x2, 2
 * x6 - x5 * x2 - x2 + x7 * x5 - x7, x2^3 * x6^2 - x7 * x2^2 * x6^2 - x7^2
 * x2 * x6^2 + 4 * x2 * x6^2 + x7^3 * x6^2 + 4 * x7 * x6^2 - 8 * x7 * x2 *
x6 - 8 * x7^2 * x6 + 8 * x7^2 * x2 ]]
premas( x1, [ x1, x3 * x2 - x2 + x7 * x3 + x7, x2 * x4 + x7 * x4 - 2 * x7 *
x2, x5 * x2 + x2 + x7 * x5 - x7, x6 * x2 - 2 * x7 * x2 + x7 * x6 ], [ x1,
x3, x4, x5, x6 ]);
0
premas( x1, [ x1 * x2 * x6 - x2 * x6 - x7 * x1 * x6 - x7 * x6 + 2 * x7 *
x2, x3 * x2^2 * x6 - x2^2 * x6 - x7^2 * x3 * x6 - x7^2 * x6 - x7 * x3 * x2^
2 + x7 * x2^2 + x7^2 * x3 * x2 + x7^2 * x2, x4 * x2 * x6 - x7 * x2 * x6 +
x7 * x4 * x6 - x7 * x4 * x2, 2 * x6 - x5 * x2 - x2 + x7 * x5 - x7, x2^3 *
x6^2 - x7 * x2^2 * x6^2 - x7^2 * x2 * x6^2 + 4 * x2 * x6^2 + x7^3 * x6^2 +
4 * x7 * x6^2 - 8 * x7 * x2 * x6 - 8 * x7^2 * x6 + 8 * x7^2 * x2 ], [ x1,
x3, x4, x5, x6 ]);
x7 * x6 * x2 - x6 * x2 + 2 * x8 * x2 - x8 * x6
init( x1, [ x1, x3, x4, x5, x6 ]);
1
init( x3 * x2 - x2 + x7 * x3 + x7, [ x1, x3, x4, x5, x6 ]);
x2 + x7
init( x2 * x4 + x7 * x4 - 2 * x7 * x2, [ x1, x3, x4, x5, x6 ]);
x2 + x7
init( x5 * x2 + x2 + x7 * x5 - x7, [ x1, x3, x4, x5, x6 ]);
x2 + x7
init( x6 * x2 - 2 * x7 * x2 + x7 * x6, [ x1, x3, x4, x5, x6 ]);
x2 + x7
```

问题形式 7 的程序

```
f1:= x6 - x2 + x1 * x6 - x5 * x2;
f1:= x6 - x2 + x1 * x6 - x5 * x2
f2:= - x4 + x2 + x1 * x4 - x3 * x2;
f2:= - x4 + x2 + x1 * x4 - x3 * x2
f3:= x7 * x4 - x8 * x3 + x4 - x8;
f3:= x7 * x4 - x8 * x3 + x4 - x8
f4:= x7 * x6 - x8 * x5 - x6 + x8;
f4:= x7 * x6 - x8 * x5 - x6 + x8
f5:= x5^2 + 2 * x5 + x6^2 - x3^2 + 2 * x3 - x4^2;
f5:= x5^2 + 2 * x5 + x6^2 - x3^2 + 2 * x3 - x4^2
f6:= 4 * x1 + 4 * x7 + 4 * x7^2 * x1 + 4 * x7 * x1^2 + 4 * x7 * x2^2 + 4 *
x8^2 * x1;
f6:= 4 * x1 + 4 * x7 + 4 * x7^2 * x1 + 4 * x7 * x1^2 + 4 * x7 * x2^2 + 4 *
x8^2 * x1
wsolve( [ f1, f2, f3, f4, f5, f6 ], [ x1, x3, x4, x5, x6, x7 ], [ ]);
[[ x1, x3 * x2 - x2 + x8 * x3 + x8, x4 * x2 - 2 * x8 * x2 + x8 * x4, x5 *
x2 + x2 + x8 * x5 - x8, x2 * x6 + x8 * x6 - 2 * x8 * x2, x7 ]]
premas( x7, [ x1, x3 * x2 - x2 + x8 * x3 + x8, x4 * x2 - 2 * x8 * x2 + x8 *
x4, x5 * x2 + x2 + x8 * x5 - x8, x2 * x6 + x8 * x6 - 2 * x8 * x2, x7 ],
[ x1, x3, x4, x5, x6, x7 ]);
0
init( x1, [ x1, x3, x4, x5, x6, x7 ]);
1
init( x3 * x2 - x2 + x8 * x3 + x8, [ x1, x3, x4, x5, x6, x7 ]);
x2 + x8
init( x4 * x2 - 2 * x8 * x2 + x8 * x4, [ x1, x3, x4, x5, x6, x7 ]);
x2 + x8
init( x5 * x2 + x2 + x8 * x5 - x8, [ x1, x3, x4, x5, x6, x7 ]);
x2 + x8
init( x2 * x6 + x8 * x6 - 2 * x8 * x2, [ x1, x3, x4, x5, x6, x7 ]);
x2 + x8
init( x7, [ x1, x3, x4, x5, x6, x7 ]);
1
```

问题形式 12 的程序

```
f1:= x6 - x2 + x1 * x6 - x5 * x2;
f1:= x6 - x2 + x1 * x6 - x5 * x2
f2:= - x4 + x2 + x1 * x4 - x3 * x2;
f2:= - x4 + x2 + x1 * x4 - x3 * x2
f3:= x7 * x4 - x8 * x3 + x4 - x8;
f3:= x7 * x4 - x8 * x3 + x4 - x8
f4:= x7 * x6 - x8 * x5 - x6 + x8;
f4:= x7 * x6 - x8 * x5 - x6 + x8
f5:= x5^2 + 2 * x5 + x6^2 - x3^2 + 2 * x3 - x4^2;
f5:= x5^2 + 2 * x5 + x6^2 - x3^2 + 2 * x3 - x4^2
f6:= 4 * x7^2 * x1 + 4 * x7 * x1^2 + 4 * x7 * x2^2 + 4 * x8^2 * x1 + 4 * x1
 + 4 * x7;
f6:= 4 * x7^2 * x1 + 4 * x7 * x1^2 + 4 * x7 * x2^2 + 4 * x8^2 * x1 + 4 * x1
 + 4 * x7
wsolve([f1,f2,f3,f4,f5,f6],[x1,x3,x4,x5,x6,x7],[]);
[[x1,x3 * x2 - x2 + x8 * x3 + x8,x2 * x4 + x8 * x4 - 2 * x8 * x2,x5 *
 x2 + x2 + x8 * x5 - x8,x6 * x2 - 2 * x8 * x2 + x8 * x6,x7]]
premas(x7,[x1,x3 * x2 - x2 + x8 * x3 + x8,x2 * x4 + x8 * x4 - 2 * x8 *
 x2,x5 * x2 + x2 + x8 * x5 - x8,x6 * x2 - 2 * x8 * x2 + x8 * x6,x7],
 [x1,x3,x4,x5,x6,x7]);
0
init(x1,[x1,x3,x4,x5,x6,x7]);
1
init(x3 * x2 - x2 + x8 * x3 + x8,[x1,x3,x4,x5,x6,x7]);
x2 + x8
init(x2 * x4 + x8 * x4 - 2 * x8 * x2,[x1,x3,x4,x5,x6,x7]);
x2 + x8
init(x5 * x2 + x2 + x8 * x5 - x8,[x1,x3,x4,x5,x6,x7]);
x2 + x8
init(x6 * x2 - 2 * x8 * x2 + x8 * x6,[x1,x3,x4,x5,x6,x7]);
x2 + x8
init(x7,[x1,x3,x4,x5,x6,x7]);
1
```

Research on four developed conjectures about Steiner-Lehmes theorem

LIAO Wenfang[1] XU Mingchun[1]

1 School of Mathematical Sceince,South China Normal University,Guangzhou 510631

Abstract The famous Steiner-Lehmes theorem is to determine whether a triangle under some conditions is isosceles or not. WANG Yanhai changed the conditions into other forms and gave nine conjectures. This paper proves three of them using Wu-method and MMP software,puts forward a new conjecture and solves it. The rest of the conjectures have not been proved because the degrees of polynomials are so high,when the geometrical conditions are changed to algebraic ones,that it causes MMP to operate too long a time and therefore cannot obtain the expected result.

Key words Steiner-Lehmes theorem;Wu-method;MMP software